Protocol Construction and Parameter Selection for Supersingular Elliptic Curve Isogeny-based Cryptography

A thesis submitted for the degree of
Doctor of Philosophy in Computer Science

by

Bruno Sterner

Surrey Centre of Cyber Security
Department of Computer Science
School of Computer Science and Electrical Engineering
Faculty of Engineering and Physical Sciences
University of Surrey

Principle Supervisor: Dr. Robert Granger
Co-supervisor: Prof. Liqun Chen

Declaration of Originality

This thesis and the work to which it refers are the results of my own efforts. Any ideas, data, images or text resulting from the work of others (whether published or unpublished) are fully identified as such within the work and attributed to their originator in the text, bibliography or in footnotes. This thesis has not been submitted in whole or in part for any other academic degree or professional qualification. I agree that the University has the right to submit my work to the plagiarism detection service TurnitinUK for originality checks. Whether or not drafts have been so-assessed, the University reserves the right to require an electronic version of the final document (as submitted) for assessment as above.

Bruno Sterner
June 2023

Acknowledgements

When reflecting on the journey that I have undertaken during my time as a PhD student, one can only think about the evolution of me as a human being. This evolution consists of many different layers and I realise that each of these layers would not have been possible without the support of so many people. I would like to take this opportunity to recognise the individuals that have enhanced my evolution during my PhD.

Firstly, I am indebted to my supervisors Robert Granger and Liqun Chen for providing me with constant support and reassurance as well as having fruitful discussions during the last few years. Their mentorship has allowed me to develop valuable skills not only as a researcher but also as a person. I will always appreciate the thoroughness that Robert Granger undertook in every single endeavour that we did together: whether it be discussions about a particular idea or comments on some written work I had done. Additionally, I am incredibly thankful to Daniel Gardham who supported me and my endeavours during my time at Surrey. On top of the many questions I asked him during my first year, we engaged in many fruitful discussions eventually leading to a collaboration on updatable public-key encryption.

I would like to thank Maria Corte-Real Santos whom I first met at the isogeny workshop in Birmingham in 2022. Having had some subsequent discussions after that event about some ideas that we had, she invited me to be part of the “smooth sandwiches" crew which consisted of Craig Costello and Jonathan Komada Eriksen, on top of myself and Maria. This crew lead to a fruitful collaboration on the problem of finding twin-smooth integers. I additionally want to thank Craig Costello for taking the time to read an early draft of parts of the thesis. His comments were extremely valuable towards the thesis itself.

The office which I have resided over the four years at Surrey, 25BB02, has been a blessing for me. Everyone that has worked in this office during this time have been wonderfully support of me and everyone else in the office. Not only did we discuss our work with each other but also amused ourselves with either chess or maths and cryptographic puzzles. On top of this, the whole department has been very supportive and created a friendly and inclusive atmosphere which has made working much easier. In particular, I want to thank the following people within the department: Catalin, Costin, Eleni, Kent, Ksenia, Mike, Nick and Rhys.

Finally and most importantly, I want to thank my entire family (from both the UK and Sweden) as well as my lifelong friends for their constant guidance and support. They are the ones who know the true evolution of me as a human being and I hope that the presentation of this thesis has made you proud of my achievements.

Abstract

Nearly all of public-key cryptosystems that are being used for real-world applications are not secure against quantum adversaries. With the fast-track development of quantum computers over the last few years there is an ever growing need to develop new cryptosystems that are believed to be resistant to quantum adversaries. The field of post-quantum cryptography aims to explore the construction and security of these cryptographic systems and has gained attraction over the last few years.

Among the various known approaches for constructing post-quantum cryptosystems is one based on isogenies between elliptic curves. The fundamental idea is that the problem of finding an isogeny between two elliptic curves is believed to be hard for both classical and quantum adversaries. Isogeny-based cryptosystems attain their security assuming the hardness of such isogeny problems. Certain cryptosystems require a more constrained isogeny problem in order to prove their security. These constrained problems are tasked to find an isogeny that has some additional structure. We note that some of these constrained problems have proved detrimental to the longevity of certain cryptosystems.

In this thesis we identify cryptographic protocols that either do not have an instantiations based on isogenies or could do with an improvement. This not only includes novel constructions of advanced protocols but also includes contributing to parameter generation of existing protocols. In particular we demonstrate novel constructions of a commitment scheme and an updatable public-key encryption scheme. We prove the security of these constructions based on either well studied isogeny problems that are believed to be hard or based on novel well motivated assumptions. Additionally, we explore new techniques for parameters for the isogeny-based signature scheme SQISign through the problem of constructing twin-smooth integers. This is a fundamental problem to study when searching for suitable parameters. To date, existing SQISign parameters only attain 128 bits of classical security. With our techniques, we are able to find parameters that attain 192 and 256 bits of classical security.

Contents

List of Figures

List of Tables

Abbreviations

AES	Advanced Encryption Standard
ARKG	Asynchronous Remote Key Generation
CM	Complex Multiplication
CSIDH	Commutative Supersingular Isogeny Diffie-Hellman
DLP	Discrete Logarithm Problem
ECDH	Elliptic Curve Diffie-Hellman
ECC	Elliptic Curve Cryptography
ECDLP	Elliptic Curve Discrete Logarithm Problem
GRH	Generalised Riemann Hypothesis
IND-CCA	Indistinguishability Against Chosen Ciphertext Attack
IND-CPA	Indistinguishability Against Chosen Plaintext Attack
NIST	National Institute of Standards in Technology
PKC	Public-Key Cryptography
PKE	Public-Key Encryption
PPT	Probabilistic Polynomial-Time
PQC	Post Quantum Cryptography
PRF	Psuedo-Random Function
ROM	Random Oracle Model
RSA	Public-key encryption scheme attributed to Rivest, Shamir, Adleman
SIDH	Supersingular Isogeny Diffie Hellman Hellman
SIKE	Supersinugular Isogeny Key Encapsulation
SQISign	Short Quarternion Isogeny Signature
UPKE	Updatable Public-Key Encryption
vOW	Algorithm attributed to van-Oorschot and Wiener

Chapter I

Introduction

Before the 1970s, cryptography was based on shared secret keys, the idea being that two parties, let us call them Alice and Bob, perform cryptographic operations using a pre-shared key. Depending on the specific circumstances, establishing this shared key can be difficult to achieve in a secure manner. At that time there would typically be some trusted third party in order to handle this exchange. Requiring such a trusted party is not necessarily the best option. This type of cryptography is commonly referred to as *symmetric cryptography* whereby a single key is used by both parties to do their respective cryptographic operations.

In the 1970s, this idea would be built on with the emergence of *public-key cryptography* (also referred to as *asymmetric cryptography*). Here the idea is that each party possesses a public and private-key pair. The public-key is made publicly available and the private-key is kept private. If Alice is the owner of a particular public-key pair, then she would do cryptographic operations with both her public and private-key while Bob will solely use her public-key if he wishes to establish communication with Alice.

The security of a vast array of modern public-key schemes are based on computationally hard problems in number theory. In other words if you believe that some problem in number theory is computationally infeasible to solve with any amount of computation time, then the underlying cryptosystem is secure. Some of these computational problems include the “integer factorisation problem" and the “discrete logarithm problem" which forms the basis of many cryptosystems used in the real world including RSA [RSA78] and ElGamal [ElG85]. The evolution of public-key cryptography has not eliminated the need for secret-key based cryptography, but it seeks to enhance it with the exploitations of public-key cryptography. For instance the classic Diffie Hellman protocol [DH76], is a trustless solution to the problem of establishing a shared secret key as mentioned previously. This Diffie Hellman protocol is an example of what is commonly referred to as a non-interactive key-exchange protocol.

Over the last five decades, public-key cryptography has opened the doors to more advanced cryptographic protocols. Each of which was designed to serve a specific application. One such idea has already been mentioned in the previous paragraph regarding non-interactive key exchange. Additionally, many of these advanced protocols that were originally of theoretical interest have featured in real world applications.

Public-key cryptography as we know it is going through a transformation. The number-theoretic problems of factoring and computing discrete logarithms that were conjectured to be resistant to classical adversaries are actually not resistant to quantum adversaries. In essence a large enough sized quantum computer could efficiently solve these number-theoretic problems and thus make the underlying cryptosystem obsolete. This is in large part due to the celebrated work of Shor [Sho94]. Modern symmetric-based cryptosystems such as AES [DR01] also suffer from quantum attacks, but these are not as devastating compared to their public-key counterparts. This is largely due to Grover's algorithm [Gro96] which takes an input to a black box function that has a specific output using $O(\sqrt{N})$ evaluations where N is the size of the domain of the function. Classical computations require $O(N)$ evaluations to produce such a output.

As a result there is a need to develop new public-key cryptosystems that are believed to be secure against both classical and quantum adversaries. The field of post-quantum cryptography is the study of such cryptosystems both from a constructive perspective but also a cryptanalytic perspective. Since 2016, the National Institute of Standards and Technology (NIST) has worked on standardising post-quantum cryptography especially as the advancement of quantum computers gets stronger [The16]. The main focus of this standardisation effort is on the selection of public-key encryption, key encapsulation mechanisms and digital signatures which are the most foundational and important cryptosystems that need replacing. Over the past year, a partial conclusion to this standardisation effort has been announced with the selection of several candidates for standardisation - see [The22]. Broadly speaking, there are six categories of post-quantum cryptosystems that are or have been considered in this standardisation process. Each category is based on either a hardness assumption or uses symmetric primitives as a building block for new primitives with post-quantum security. These categories are summarised below.

- Code-based cryptography [McE78,ABB+17]: builds cryptosystems that uses error-correcting codes such as Goppa codes. Well known cryptosystems include McEliece and BIKE – both of which are still in the NIST standardisation process. Pros: the security assumptions have been studied for close to five decades, so the confidence in its security is strong. Cons: public-key sizes are quite large;

- Hash-based cryptography [BHH+15]: builds cryptographic protocols from the security of hash functions. SPHINCS+ has been selected for standardisation by NIST. Pros: There is a strong confidence in its security. Cons: key sizes are quite large;

- Multivariate-based cryptography [DS05]: based on the hard problem of solving a set of multivariate polynomial equations. Pros: reasonable signature sizes and performs well in practice. Cons: confidence in its security is low at the moment;

- Lattice-based cryptography [BDK+18, DKL+18, FHK+18]: build cryptosystems based on hard problems using lattices such as shortest vector problem and bounded distance decoding to name a few. A number of lattices-based schemes have been selected for standarisation by NIST including Kyber, Dilithium and Falcon. Pros: Most well rounded when it comes to key sizes, performance and security. Cons: some security assumptions have been questioned and so over reliance on lattices may not be well advised;

- Isogeny-based cryptography [JAC+17]: builds cryptosystems based on computing isogenies of elliptic curves. Numerous protocols have been designed that resemble classical public-key protocols. Pros: shortest key sizes among all categories. Cons: orders of magnitude slower than other candidates;

- MPC-based cryptography [CDG$^+$17]: uses symmetric primitives such as psuedo random functions combined with a multi-party computation to build cryptographic primitives. Pros: the use of certain symmetric primitives give rise to short signatures. Cons: needs to stand the test of time in order to gain confidence in its security.

We also note that there are other bodies around the world that are attempting to standardise post-quantum cryptography [ETS].

In this work we explore isogeny-based cryptography as an avenue to construct post-quantum cryptosystems. As mentioned in the above summary, the field of isogeny-based cryptography builds cryptosystems based on computing isogenies between elliptic curve. It relies on the fact that given two elliptic curves for which there exists an isogeny between them, finding an isogeny is conjectured to be hard for both classical and quantum adversaries. We note that by Tate's theorem [Tat66], there are polynomial-time algorithms to check whether there exists an isogeny between two elliptic curves. It can be thought of as a quantum resistant alternative to classical elliptic curve cryptography (ECC) which for the most part relies on the hardness of computing discrete logarithms within the group of points on an elliptic curve and has seen numerous real world applications.

Isogeny-based cryptography started in 1997 thanks to Couveignes through an unpublished manuscript [Cou06]. His idea was not made publicly available until it was independently rediscovered by Rostovtsev and Stolbunov [RS06]. They were the first to consider to possibility of isogeny-based cryptosystems as a candidate to achieve post-quantum security. These initial cryptosystems used ordinary elliptic curves owing in large part to the widespread use of such curves in traditional ECC. In more recent years supersingular elliptic curves have paved the way for more efficient cryptosystems that also attain stronger security. The most notable cryptosystems based on supersingular curves are: the CGL hash function [CLG09], the SIDH key exchange protocol [JDF11], the CSIDH key exchange protocol [CLM$^+$18] and the SQISign digital signature [DFKL$^+$20]. In particular, the key exchange protocol SIDH formed the basis for a submission to the NIST standardisation process called SIKE [JAC$^+$17]. The security of SIKE relies on a slightly modified isogeny problem whereby some extra auxillary information is given. Over the years, this problem has received a huge amount of scrutiny - ultimately resulting in the protocol being completely broken and hence is no longer in the NIST standardisation process. The other protocols mentioned above still remain secure against both classical and quantum adversaries.

Despite the effort from the standardisation bodies to standardise post-quantum cryptography, the focus of their endeavours has been on the standardisation of these foundational cryptosystems. As a result, a majority of the research community has focussed on the construction and analyses of these foundational cryptosystems. While these will be important to set the foundations of real-world post-quantum cryptography over the coming years, there are more advanced cryptosystems that have real-world value and require a post-quantum solution. Over the last few years, this area of research has been gaining more attention not only from the perspective of post-quantum cryptography as a whole, but also for isogeny-based cryptography. In this thesis, we aim to construct or improve upon existing cryptosystems based on isogenies. This not only includes exploring the foundational cryptosystems, but also exploring some advanced protocols.

Thesis structure and the author's main contributions. The work presented here is in two parts. Part 1 introduces the necessary cryptographic and mathematical background that is used in the work that follows, while Part 2 details our contributions to the field of isogeny-based cryptography.

In Chapter II we give an introduction to the fundamental cryptographic primitives that have played a key role within the advancement of isogeny-based cryptography. Most notably this includes public-key encryption and digital signatures which have been the focus of the NIST standardisation effort. In Chapter III we introduce the necessary mathematical background that is relevant for the thesis. This not only includes a study of elliptic curves and isogenies but also other number theoretic tools which are used in the thesis. In Chapter IV we showcase the published and ongoing work that encompasses isogeny-based cryptography. This includes looking at algorithmic tools for isogenies as well as presenting some well-known cryptosystems.

The focus of Chapter V is on work that was presented at Mathcrypt 2021 and published in a special issue of the Journal of Mathematical Cryptology [Ste21]. Concretely, we define the notion of commitment schemes and formalise how one can construct such a scheme based on walks in supersingular isogeny graphs. This is something that, prior to this work, had not been achieved in the literature. The idea behind the commitment scheme is based on the well-known and generic construction based on hash functions but formalising the security in a way that does not require any random oracles. In Chapter VI we present an extension of the work from [BSC$^+$22] which has been has been accepted to Asiacrypt23. The focus of the chapter is on finding parameters for the isogeny-based signature scheme SQISign which is discussed in Chapter IV. At the heart of these parameters, one requires a prime, p, of cryptographic size such that $p^2 - 1$ has a large smooth cofactor. We develop a new method for finding SQISign friendly primes based on the evaluation of certain polynomials. Inputs to these polynomials are found using an algorithm, referred to as CHM, that finds twin-smooth integers. These polynomials not only include that classical polynomials, $p_n(x) = 2x^n - 1$, that has been used in prior works on this topic but also new polynomials that are particularly favourable in this setting. In Chapter VII we present work from [GS23]. In it we explore the construction of an updatable public-key encryption scheme based on cryptographic group actions. While other updatable public-key encryption schemes based on such group actions have been presented in the literature, none have been designed with the intent of applying it to the specific setup within CSIDH. Our scheme achieves this property. To achieve this, we provide a construction of an asychronous key generation scheme that is then transformed into an updatable public-key encryption with the aid of a public-key encryption scheme.

Finally, in Chapter VIII, we give a summary of the core contributions that have been made in this thesis as well as layout some interesting open problems that could serve as future work.

Part 1

Preliminaries

Chapter II

Modern Cryptography

We begin by introducing some of the important notions and primitives that are seen in modern day cryptography and are also fundamental to many applications. As one would imagine the potential scope of primitives to dive into is vast. So in this chapter, we start off by looking into primitives that have been central to the development of isogeny-based cryptography. This includes cryptographic hash functions, non-interactive key exchange, public-key encryption and signature schemes – the specific applications built from isogenies will be discussed in Chapter IV. Later chapters other cryptographic primitives will be introduced. These will be introduced to serve the purpose of the contribution of said chapter. For a wider scope of primitives to explore, we refer the reader to books by Katz and Lindell [KL20] and Smart [Sma15].

Definition II.1 (Negligible function). *Let $\epsilon : \mathbb{N} \to \mathbb{R}_{>0}$ be function. We say that ϵ is negligible if for every polynomial, p, there is an integer $N \in \mathbb{N}$ such that for all $n \geq N$ we have*

$$\epsilon(n) \leq \frac{1}{p(n)}.$$

Throughout this text, a negligible function will often be written as $\mathrm{negl}(\cdot)$ *(or simply* negl*).*

A probabalistic polynomial-time (PPT) algorithm with respect to a parameter n, is an algorithm such that the expected run time is polynomial in n. In other words, it runs in $O(n^k)$ for some positive integer k. Throughout this thesis, there will be an implicit input 1^λ in each of the algorithms presented. Here λ is called a security parameter and refers to a measure of security of a cryptographic protocol. An adversary, which will be typically be denoted by $\mathscr{A}$, is a PPT algorithm with respect to a security parameter λ.

II.1 Symmetric Cryptography

In this section we discuss the symmetric primitives that are foundational to some of the isogeny-based applications. In particular we discuss cryptographic hash functions.

```
$\mathsf{Exp}^{\mathsf{pre-image}}_{\mathscr{H},y,\mathscr{A}}(\lambda)$
1 : $\mathsf{pp} \leftarrow \mathsf{KeyGen}()$
2 : $x \leftarrow \mathscr{A}(\mathsf{pp}, \mathscr{H}, y)$
3 : $\textbf{return } 1 \textbf{ if } \mathscr{H}(x) = y \textbf{ else } 0$
```

```
$\mathsf{Exp}^{\mathsf{2nd-pre-image}}_{\mathscr{H},x,\mathscr{A}}(\lambda)$
1 : $\mathsf{pp} \leftarrow \mathsf{KeyGen}()$
2 : $x' \leftarrow \mathscr{A}(\mathsf{pp}, \mathscr{H}, x)$
3 : $\textbf{return } 1 \textbf{ if } \mathscr{H}(x) = \mathscr{H}(x') \textbf{ else } 0$
```

```
$\mathsf{Exp}^{\mathsf{collision}}_{\mathscr{H},\mathscr{A}}(\lambda)$
1 : $\mathsf{pp} \leftarrow \mathsf{KeyGen}()$
2 : $(x, x') \leftarrow \mathscr{A}(\mathsf{pp}, \mathscr{H})$
3 : $\textbf{return } 1 \textbf{ if } \mathscr{H}(x) = \mathscr{H}(x') \textbf{ else } 0$
```

Figure 1: The pre-image, second pre-image and collision resistance experiments for a cryptographic hash function.

II.1.1 Cryptographic Hash Functions

A hash function is a deterministic map that takes as input data of arbitrary size and outputs a fixed length bit string. One can informally turn these hash functions into a cryptographic hash function by imposing a “one-way" property on it. In other words it is computationally infeasible to invert the hash function. In addition, we also need the function to be resistant to collisions – so it is computationally infesible to provide two inputs that hash to the same output.

More formally, a cryptographic hash function consists of two algorithms: $\mathsf{KeyGen}()$ and $\mathscr{H}(\cdot)$. $\mathsf{KeyGen}()$ is a PPT algorithm that outputs the necessary public parameters and $\mathscr{H}(\cdot)$ is a deterministic algorithm that takes as input a value x and outputs the result $\mathscr{H}(x) \in \{0,1\}^{\lambda}$. We formally define the necessary security properties for these hash functions. Throughout these upcoming definitions, we let $\mathscr{H}$ is a cryptographic hash function and $\mathscr{A}$ is a PPT adversary.

Definition II.2 (Pre-image resistance). *The pre-image resistance advantage for an adversary $\mathscr{A}$, denoted by $\mathrm{Adv}^{\textit{pre-image}}_{\mathscr{H}}(\mathscr{A})$, is defined to be*

$$\Pr\left[\mathsf{Exp}^{\mathsf{pre-image}}_{\mathscr{H},y,\mathscr{A}}(\lambda) = 1\right].$$

where $\mathsf{Exp}^{\mathsf{pre-image}}_{\mathscr{H},y,\mathscr{A}}(\lambda)$ is the pre-image resistant experiment as defined in Figure 1. We say that the hash function $\mathscr{H}$ is preimage resistant if, for every adversary $\mathscr{A}$, there is a negligible function, negl, *such that the advantage is bounded above by* $\mathrm{negl}(\lambda)$.

Definition II.3 (Second pre-image resistance). *The second pre-image resistance advantage for an adversary $\mathscr{A}$, denoted by $\mathrm{Adv}^{\textit{2nd-pre-image}}_{\mathscr{H}}(\mathscr{A})$, is defined to be*

$$\Pr\left[\mathsf{Exp}^{\mathsf{2nd-pre-image}}_{\mathscr{H},x,\mathscr{A}}(\lambda) = 1\right].$$

where $\mathsf{Exp}^{\mathsf{2nd-pre-image}}_{\mathscr{H},y,\mathscr{A}}(\lambda)$ is the second pre-image resistant experiment as defined in Figure 1. We say that the hash function $\mathscr{H}$ is second preimage resistant if, for every adversary $\mathscr{A}$, there is a negligible function, negl, *such that the advantage is bounded above by* $\mathrm{negl}(\lambda)$.

Definition II.4 (Collision resistance). *The collision resistance advantage for an adversary $\mathscr{A}$, denoted by $\mathrm{Adv}^{\textit{collision}}_{\mathscr{H}}(\mathscr{A})$, is defined to be*

$$\Pr\left[\mathsf{Exp}^{\mathsf{collision}}_{\mathscr{H},\mathscr{A}}(\lambda) = 1\right].$$

where $\mathsf{Exp}^{\mathsf{collision}}_{\mathscr{H},y,\mathscr{A}}(\lambda)$ *is the collision resistant experiment as defined in Figure 1. We say that the hash function* $\mathscr{H}$ *is collision resistant if, for every adversary* $\mathscr{A}$*, there is a negligible function,* negl*, such that the advantage is bounded above by* $\mathrm{negl}(\lambda)$*.*

The Random oracle model. In an ideal world, one would like to prove the security of a cryptographic protocol in the *standard model*. This model does not make any assumptions on any aspect of the underlying cryptography that could be deemed as suspicious. However, in many security proofs of cryptographic protocols, one requires a strong randomness assumptions about the outputs of certain functions which includes hash functions. The *random oracle model* (ROM) is a model that allows one show the security of a protocol holds if an attacker could understand the behaviour of such a random looking function. Cryptographic hash functions functions are common examples of functions that can be modelled as a random oracle.

II.2 Asymmetric Cryptography

In this section we discuss the asymmetric (also referred to as public-key) primitives that are foundational to some of the isogeny-based applications. In particular we discuss non-interactive key exchange, public-key encryption and digital signature schemes.

II.2.1 Non-Interactive Key Exchange

A non-interactive key exchange protocol is an technique to establish a shared secret between two parties in a way that requires no interaction between the parties. The basic framework for such a protocol involves each party establishing a public-key and private-key pair and once the respective public key is revealed, the other party can perform computations on the public key in such a way to establish a shared secret. The idea of such a scheme dates back to the classical Diffie-Hellman protocol [DH76]. In this scheme, exponentiation in a finite group is used in order to obtain a shared secret. Recently, formal security models have been propsed for such protocols [CK01, FHKP13].

II.2.2 Public-Key Encryption

A public-key encryption scheme allows participants to encrypt a message using the recipient's public key. Decrypting a ciphertext can only be done with the possession of a corresponding secret key which is stored secretly by the owner of the public key.

More formally, a public-key encryption scheme (PKE) consists of three algorithms: $\mathsf{KeyGen}()$, $\mathsf{Enc}()$ and $\mathsf{Dec}()$. $\mathsf{KeyGen}()$ is an algorithm that returns the necessary public parameters along with a key pair $(\mathrm{sk}, \mathrm{pk})$. $\mathsf{Enc}()$ is an algorithm that takes as input a public key pk, a plaintext m along with some (optional) randomness r and outputs a ciphertext c. $\mathsf{Dec}()$ is an algorithm that takes as input a ciphertext c along with a secret key sk and outputs a plaintext m.

The correctness of a PKE scheme is a property that requires that the ciphertext produced with pk for a plaintext m should either always decrypt to m using sk or do so with extremely high probability. Informally, the one-way security of a PKE scheme says that it should be hard to take a ciphertext and recover the plaintext without the possession of a secret key. There is a

```
$\mathsf{Exp}^{\mathsf{IND\text{-}CPA}}_{\mathsf{PKE},\mathscr{A}}(\lambda)$
1 : $(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{KeyGen}()$
2 : $(m_0, m_1) \leftarrow \mathscr{A}$
3 : $b \leftarrow\!\$\ \{0,1\}$
4 : $c \leftarrow \mathsf{Encrypt}(\mathsf{pk}, m_b)$
5 : $b' \leftarrow \mathscr{A}(c)$
6 : return $b \overset{?}{=} b'$
```

```
$\mathsf{Exp}^{\mathsf{IND\text{-}CCA}}_{\mathsf{PKE},\mathscr{A}^{\mathcal{O}_{\mathsf{dec}}}}(\lambda)$
1 : $(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{KeyGen}()$
2 : $(m_0, m_1) \leftarrow \mathscr{A}^{\mathcal{O}_{\mathsf{dec}}}$
3 : $b \leftarrow\!\$\ \{0,1\}$
4 : $c \leftarrow \mathsf{Encrypt}(\mathsf{pk}, m_b)$
5 : $b' \leftarrow \mathscr{A}^{\mathcal{O}_{\mathsf{dec}}}(c)$
6 : return $b \overset{?}{=} b'$
```

```
$\mathcal{O}_{\mathsf{dec}}(c^*, c)$
1 : assert $c^* \neq c$
2 : $m \leftarrow \mathsf{Decrypt}(\mathsf{sk}, c)$
3 : return $m$
```

Figure 2: The IND-CPA *and* IND-CCA *experiements for a public key encryption scheme as well as the decryption oracle $\mathcal{O}_{dec}$. For the* IND-CCA *experiment, the adversary $\mathscr{A}^{\mathcal{O}_{\mathsf{dec}}}$ is given access to the decryption oracle $\mathcal{O}_{dec}$.*

stronger notion of security of a PKE scheme. The issue with one-way security is that it does not cover the possibility of recovering partial information about the plaintext. This is rectified by introducing *indistinguishability games*. The idea is that one is given the ciphertext of one of two messages and you are tasked to recover which message was used to get the ciphertext. This is formalised this through the upcoming definitions and we write $\mathsf{PKE} = (\mathsf{KeyGen}(), \mathsf{Enc}(), \mathsf{Dec}())$ for a public-key encryption scheme.

Definition II.5. *The* indistinguishability under chosen-plaintext attack (IND-CPA) *advantage for an adversary $\mathscr{A}$, denoted by* $\mathrm{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathrm{PKE},\mathscr{A}}(\lambda)$, *is defined to be*

$$2\left|\Pr\left[\mathsf{Exp}^{\mathsf{IND\text{-}CPA}}_{\mathsf{PKE},\mathscr{A}}(\lambda) = 1\right] - \frac{1}{2}\right|.$$

where $\mathsf{Exp}^{\mathsf{IND\text{-}CPA}}_{\mathsf{PKE},\mathscr{A}}(\lambda)$ *is the* IND-CPA *experiment as defined in Figure 2. We say that the public-key encryption scheme* PKE *is* IND-CPA*-secure if, for every adversary $\mathscr{A}$, there is a negligible function,* negl*, such that this* IND-CPA *advantage is bounded above by* $\mathrm{negl}(\lambda)$.

In the following slightly stronger definition of security, the adversary is given access to a *decryption oracle* $\mathcal{O}_{\mathsf{dec}}$. The idea is that on top of the knowledge of the ciphertext of either m_0 or m_1 the adversary can query the ciphertext of other plaintexts m that differ from m_0 and m_1 – using this information to see if it can learn anything about the challenged ciphertext.

Definition II.6. *The* indistinguishability under chosen-ciphertext attack (IND-CCA) *advantage for an adversary $\mathscr{A}^{\mathcal{O}_{\mathsf{dec}}}$ that has access to a decryption oracle as defined in Figure 2, denoted by* $\mathrm{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathrm{PKE},\mathscr{A}}(\lambda)$, *is defined to be*

$$2\left|\Pr\left[\mathsf{Exp}^{\mathsf{IND\text{-}CCA}}_{\mathsf{PKE},\mathscr{A}^{\mathcal{O}_{dec}}}(\lambda) = 1\right] - \frac{1}{2}\right|.$$

where $\mathsf{Exp}^{\mathsf{IND\text{-}CCA}}_{\mathsf{PKE},\mathscr{A}^{\mathcal{O}_{dec}}}(\lambda)$ *is the* IND-CCA *experiment as defined in Figure 2. We say that the public-key encryption scheme* PKE *is* IND-CCA*-secure if, for every adversary $\mathscr{A}$, there is a negligible function,* negl*, such that this* IND-CCA *advantage is bounded above by* $\mathrm{negl}(\lambda)$.

Key encapsulation mechanisms. In recent years, a key encapsulation mechanism (KEM) [Sho01,CS03] has been proposed with a goal of establishing a secret secret between two parties

```
$\mathsf{Exp}^{\mathsf{EUF-CMA}}_{\mathsf{DS},\mathscr{A}^{\mathcal{O}_{\mathtt{Sign}}}}(\lambda)$
1 : $(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{KeyGen}$
2 : $\mathsf{QueryList} \leftarrow \{m_i\}_{i=1}^{n}$
3 : $(m^*,\sigma^*) \leftarrow \mathscr{A}^{\mathcal{O}_{\mathtt{Sign}}(m_i)}(\mathsf{pk})$
4 : **assert** $m^* \notin \mathsf{QueryList}$
5 : **return** $\mathsf{Verify}(\mathsf{pk},\sigma^*,m^*)$

$\mathcal{O}_{\mathtt{Sign}}(m)$
1 : $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk},m)$
2 : **return** $\sigma$
```

Figure 3: The EUF − CMA *experiement for a digial signature scheme as well as the signing oracle* $\mathcal{O}_{\mathtt{Sign}}$.

that own a public key through the means of some encryption. One can think of these mechanisms as a key exchange protocol that has some interaction. More precisely, it consists of three algorithms: KeyGen(), Encap() and Decap(). KeyGen() is just like for a public key encryption scheme. The encapsulation algorithm Encap() takes a public key pk as input an outputs a pair (k,c) where k is a key within some key-space and c is a corresponding cipher text. The decapsulation algorithm Decap() takes as input a secret key sk and a outputs a key k'. The correctness of a KEM ensures that for all outputs of the encapsulation algorithm (k,c) on some public key pk, the application of the decapsulation algorithm on the corresponding secret key sk either always results in $k = k'$ or does so with extremely high probability. Moreover one can define IND-CPA and IND-CCA security of a KEM in a similar way as done for public-key encryption schemes.

II.2.3 Digital Signatures

A digital signature scheme allows the owner of a public key pair (pk, sk) to sign a message on his secret key sk. This produces a signature which can be sent to other parties in such a way so that along with the message the signature can authenticate the signer of the message.

More formally, a digital signature scheme consists of three algorithms: KeyGen(), Sign() and Verify(). KeyGen() is an algorithm that returns the necessary public parameters along with a key pair (sk, pk). Sign() is an algorithm that takes as input a secret key sk and a message m and outputs a signature σ. Verify() is an algorithm that takes as input a public key pk, a signature σ and a message m and outputs a boolean value $b \in \{0,1\}$ according to whether the signature σ is a valid signature of the message m on the public key pk.

The correctness of a signature scheme is a property that requires that the signature produced with sk for a message m should either always accept the verification algorithm along with pk or do so with extremely high probability. A common security notion of a signature scheme is called unforgeability. The basic idea behind it is that someone who does not own the secret key sk associated to a public key pk, should not be able to produce a valid signature of a message under that public key even if it is given a collection of message and signature pairs. This is formalised in the following definition and we write $\mathsf{DS} = (\mathsf{KeyGen}(),\mathsf{Sign}(),\mathsf{Verify}())$ for a digital signature scheme.

Definition II.7. *The* unforgeability under adaptive chosen message attack (EUF − CMA) *advantage for an adversary* $\mathscr{A}^{\mathcal{O}_{\mathtt{Sign}}}$ *that has access to a signing oracle as defined in Figure 3, denoted* *by* $\mathrm{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathrm{DS},\mathscr{A}}(\lambda)$*, is defined to be*

$$2\left|\Pr\left[\mathsf{Exp}^{\mathsf{EUF-CMA}}_{\mathrm{DS},\mathscr{A}^{\mathcal{O}_{\mathtt{Sign}}}}(\lambda)=1\right]-\frac{1}{2}\right|.$$

where $\mathsf{Exp}^{\mathsf{EUF-CMA}}_{\mathrm{DS},\mathscr{A}^{\mathcal{O}_{\mathtt{Sign}}}}(\lambda)$ *is the* $\mathsf{EUF-CMA}$ *experiment as defined in Figure 2. We say that the public-key encryption scheme* PKE *is* $\mathsf{IND\text{-}CCA}$*-secure if, for every adversary* $\mathscr{A}$*, there is a negligible function,* negl*, such that this* $\mathsf{IND\text{-}CCA}$ *advantage is bounded above by* $\mathrm{negl}(\lambda)$*.*

Identification and Sigma Protocols. An identification scheme is a protocol between a prover and a verifier in which the prover wants to prove his identity to the verifier. Sigma protocols is in some sense an abstraction of these identification protocols in which a prover wants to prove that it knows a witness to a statement within some NP-relation. In the setting of identification protocols, the witness is typically a public key and the statement is a private key. For both identification and sigma protocols, they typically consist of three stages: the prover produces a commitment which it sends to the verifier; the verifier issues a challenge which it sends to the prover; and finally the prover produces a response from the challenge which it sends to the verifier. Then the verifier either accepts or rejects the response.

The correctness of such a protocol ensures that the verifier always accepts a valid response. The soundness of such a protocol ensures the existence of an efficient extractor such that given two transcripts (or valid interactions of a prover) will output the underlying statement. Finally the zero-knowledge of such a protocols ensures that there is a simulation algorithm that takes as input a witness and a challenge and outputs a commitment and a response such that one can simulate such a protocol with these values.

Fiat-Shamir Transform. From the description of these identification and sigma protocols, there is some level of an interaction between the prover and the verifier. In particular, the interaction consists of the exchange of the commitment, challenge and response values. The Fiat-Shamir transform [FS87] is a technique that can make these identification and sigma protocols non-interactive with the inclusion of cryptographic hash functions. The idea is to simulate a seemingly random challenge by hashing the commitment value along with the witness. This transform has been used to turn an identification protocol into a signature scheme which satisfies the unforgeability condition from Definition II.7 in the random oracle model. We note that a variant of this Fiat-Shamir tranform due to Unruh [Unr15] deals with the quantum-resistant option.

Chapter III

Number Theory

In this chapter we introduce the necessary number theoretic tools and results that are foundational to the work that follows. We begin by giving a background on elliptic curves that is central to the cryptographic applications that arise from them. This also includes background on isogenies between elliptic curves which is the main topic of exploration for this thesis. We also introduce the class group and the Deuring correspondence, as well as results on smooth numbers and how they are distributed. There are many excellent resources that give a thorough introductions to all of these tools. For elliptic curves and isogenies, the books by Silverman [Sil09] and Galbraith [Gal12] are excellent resources; for the algebraic number theory and in particular class groups, we refer the reader to Stewart and Tall's book [ST15]; for quaternion algebras, a comprehensive background is given in Voight's book [Voi21]; and for the analytic number theory that underpins not only smooth numbers but also prime numbers, we refer the reader to [Ten06, Ten15] as well as some [Sho09], which addresses computation aspects.

Throughout this chapter we will use affine and projective space. *Affine space of dimension n* over a field k, written $\mathbb{A}^n(k)$, is the set n-tuples with entries in k. We write elements in this affine space as $(x_0, \ldots, x_{n-1})$. *Projective space of dimension n*, written $\mathbb{P}^n(k)$, is the set $\mathbb{A}^{n+1} \setminus \{(0, \ldots, 0)\}$ modulo the following equivalence relation $\sim$. For two elements $\mathbf{x} = (x_0, \ldots, x_n), \mathbf{y} = (y_0, \ldots, y_n) \in \mathbb{A}^{n+1} \setminus \{(0, \ldots, 0)\}$, we say $\mathbf{x} \sim \mathbf{y}$ is there is an $\alpha \in k^* := k \setminus \{0\}$ such that $x_i = \alpha y_i$ for all i. We write elements in this projective space as $[X_0 : \ldots : X_n]$. Sometimes $\mathbb{A}^n$ and $\mathbb{P}^n$ will be written instead of $\mathbb{A}^n(k)$ and $\mathbb{P}^n(k)$ when the field in question is understood.

Projective space of dimension n can be thought of as a copy of affine space of dimension n along with some additional points. For an element $\mathbf{X} \in \mathbb{P}^n$ either X_n is zero or non-zero. If $X_n \neq 0$ then, by means of the equivalence relation, we can scale this to 1 and get an affine point $\mathbf{x} \in \mathbb{A}^n$ with $x_i = X_i / X_n$. If $X_n = 0$ then we do not get an affine point and we call such a point a *point at infinity*.

The algebraic closure of a field k, is an algebraic extension K/k for which every polynomial in $K[x]$ has a root in K. One can show that an algebraic closure always exists and is unique up to isomorphism. Typically we write $\overline{k} := K$ for the algebraic closure of k. In the specific situation when k is a finite field with $q = p^n$ elements for a prime p and a positive integer n, we write $k = \mathbb{F}_q$ and $\overline{k} = \overline{\mathbb{F}_p}$ for its algebraic closure.

III.1 Elliptic Curves

A *general Weierstrass equation* over a field k is a curve in $\mathbb{P}^2(k)$ given by an equation of the form

$$E : Y^2Z + a_1XYZ + a_3YZ^2 = X^3 + a_2X^2Z + a_4XZ^2 + a_6Z^3$$

with coefficients $a_i \in k$. There is only one point at infinity on this curve[1], namely the point $\mathcal{O}_E = [0 : 1 : 0]$. So one can define such a Weierstrass equation in affine coordinates by including this additional point at infinity. From now on, unless stated otherwise, all Weierstrass equations (and other equations) will be written in affine coordinates. For a general Weierstrass equation, define the following quantities:

$$\begin{aligned} b_2 &= a_1^2 + 4a_2 \\ b_4 &= 2a_4 + a_1a_3 \\ b_6 &= a_3^2 + 4a_6 \\ b_8 &= a_1^2a_6 + 4a_2a_6 - a_1a_3a_4 \\ &\quad + a_2a_3^2 - a_4^2 \end{aligned} \qquad \begin{aligned} c_4 &= b_2^2 - 24b_4 \\ \Delta(E) &= -b_2^2b_8 - 8b_4^3 - 27b_6^2 + 9b_2b_4b_6 \\ j(E) &= \frac{c_4^3}{\Delta} \end{aligned} \tag{III.1}$$

We call the quantity $\Delta(E)$ the discriminant and the quantity $j(E)$ the j-invariant of the Weierstrass equation. These quantities will be important to the theory of elliptic curves. When the characteristic of the field k is not 2 or 3, then by applying some linear transformations we can turn a general Weierstrass equation into a *short Weierstrass equation* which, in affine coordinates, has the form: $y^2 = x^3 + Ax + B$. For short Weierstrass equations, these quantities simplify to $\Delta(E) = -16(4A^3 + 27B^2)$ and $j(E) = 1728\frac{4A^3}{4A^3+27B^2}$.

Definition III.1. *Let E/k be a general Weierstrass equation over k and $\mathcal{O}_E$ be its associated point at infinity. We say $(E, \mathcal{O}_E)$ is an* elliptic curve *if it is smooth in the sense that it's discriminant, $\Delta = \Delta(E)$, is non-zero.*

There are different models to define elliptic curves other than using the Weierstrass model. These include *Montgomery curves* [Mon87] which are defined by the equation $By^2 = x^3+Ax^2+x$ and *(twisted) Edwards curves* [Edw07,BBJ+08] which are defined by $ax^2+y^2 = 1+dx^2y^2$. These alternative models are widely used in practical implementations that use elliptic curves - due to a faster underlying arithmetic. For the purpose of this thesis, most elliptic curves will be given by a short Weierstrass equation. Unless directly stated otherwise, one can assume this is the case.

III.1.1 Group Structure

Despite the geometric definition of these elliptic curves, there is an algebraic structure that can be associated to the points on the elliptic curve. For an elliptic curve E over a field k given in general Weierstrass form and a field extension K/k, we denote by $E(K)$ the set of K-rational points on this curve including the point at infinity, namely

$$E(K) := \{(x, y) \in K^2 : y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6\} \cup \{\mathcal{O}_E\}.$$

[1]Often referred to as the distinguished point on E.

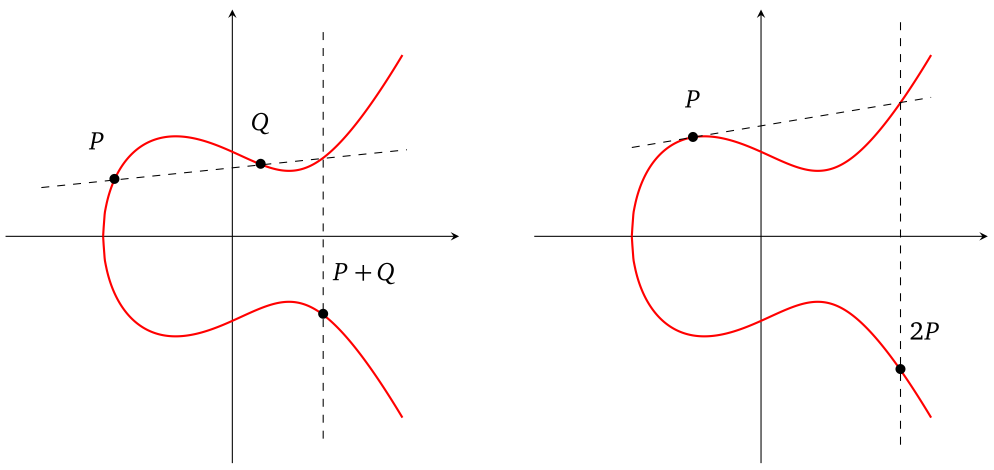


Figure 4: Visual depiction of the chord and tangent rule for point addition and point doubling on an elliptic curve over $\mathbb{R}$.

This set of K-rational points has a special addition rule, denoted by $+$, which takes as input two points in $E(K)$ and returns another point on $E(K)$. This addition rule on an elliptic curve is called the chord and tangent rule and proceeds as follows. Let $P, Q \in E(K)$ be K-rational points on the curve. If P (Q resp.) is the point at infinity, then we define $P+Q := Q$ ($P+Q := P$ resp.). Now assume that the points P, Q are not the point at infinity. Then they both have affine coordinates of the form $P = (x_0, y_0)$ and $Q = (x_1, y_1)$. If $x_0 = x_1$ and $y_0 = -y_1$, then we define $P + Q := \mathcal{O}_E$. If $x_0 = x_1$ and $y_0 = y_1$ (so $P = Q$), then set $m = (3x_0^2 + A)/2y_0$ and otherwise set $m = (y_1 - y_0)/(x_1 - x_0)$. Then define

$$P + Q := (x_2, -y_2),$$

where $x_2 = m^2 - x_0 - x_1$ and $y_2 = y_0 + m(x_2 - x_0)$. When $P = Q$, we write $2P := P + P$ and for an integer $m \geq 3$ we write $mP := (m-1)P + P = P + \cdots + P$. This is often referred to as scalar multiplication of a point on the elliptic curve. Additionally, for an affine point $P = (x_0, y_0)$, we write $-P := (x_0, -y_0)$ and, more generally, write $(-m)P := -(mP)$ for a positive integer m. For the point at infinity, we write $-\mathcal{O}_E := \mathcal{O}_E$.

Equipping the set $E(K)$ with this chord and tangent rule turns it into a group. For a detailed justification of why this forms a group, we refer the reader to [Sil09, Proposition III.2.2]. By very definition, the point at infinity acts as the identity of the group and the inverse of a point P is the point $-P$. See Figure 4 for an visual illustration of the chord and tangent rule for an elliptic curve over the real numbers.

Division polynomial. For $m \in \mathbb{Z}$, we call a point P an m-torsion point if $mP = \mathcal{O}_E$. The *multiplication-by-m map* is the map between point on E over the algebraic closure of k, $[m] : E(\overline{k}) \to E(\overline{k})$, where $P \mapsto mP$. For divisors d of m, d-torsion points lie in the kernel of this multiplication-by-m map. The m-torsion subgroup, denoted by $E[m]$, is the kernel of the multiplication-by-m map. Furthermore, for a field extension K/k, we set $E(K)[m] := E[m] \cap E(K)$ to be the subgroup of m-torsion points that are defined over the extension K. Recall the quantities b_2, b_4, b_6, b_8 as defined in Equation III.1, the m^{th}-*division polynomial*, denoted by ψ_m, is defined through the following iterative process:

$$\begin{aligned}
\psi_1(x,y) &= 1, \qquad \psi_2(x,y) = 2y + a_1x + a_3,\\
\psi_3(x,y) &= 3x^4 + b_2x^3 + 3b_4x^2 + 3b_6x + b_8,\\
\psi_4(x,y) &= \psi_2(x,y)\big(2x^6 + b_2x^5 + 5b_4x^4 + 10b_6x^3 + 10b_8x^2\\
&\qquad +(b_2b_8 - b_4b_6)x + (b_4b_8 - b_6^2)\big),\\
\psi_{2n+1} &= \psi_{n+2}\psi_n^3 - \psi_{n-1}\psi_{n+1}^3,\\
\psi_{2n} &= (\psi_{n-1}^2\psi_n\psi_{n+2} - \psi_{n-2}\psi_n\psi_{n+1}^2)/\psi_2.
\end{aligned} \tag{III.2}$$

Furthermore, set $\phi_m = x\psi_m^2 - \psi_{m+1}\psi_{m-1}$ and $\omega_m = (\psi_{m-1}^2\psi_{m+2} + \psi_{m-2}\psi_{m+1}^2)/4y$. Then for a point $P \in E(k)$, we have $mP = \left(\frac{\phi_m(P)}{\psi_m(P)^2}, \frac{\omega_m(P)}{\psi_m(P)^3}\right)$ [Sil09, Exercise 3.7]. Viewing this formula from a projective perspective, we can deduce that a point P is an m-torsion point if and only if P vanishes on the m^{th}-division polynomial, ψ_m.

When the characteristic of k is zero or the characteristic does not divide m then we have $E[m] \cong \mathbb{Z}/m\mathbb{Z} \times \mathbb{Z}/m\mathbb{Z}$ and if p is the characteristic of k then $E[p^e]$ is isomorphic to either $\mathbb{Z}/p^e\mathbb{Z}$ or $\{\mathcal{O}_E\}$ [Sil09, Corollary III.6.4]. We say E is *ordinary* if, for all exponents e, $E[p^e] \cong \mathbb{Z}/p^e\mathbb{Z}$ and E is *supersingular* if $E[p^e] \cong \{\mathcal{O}_E\}$.

III.1.2 Elliptic Curves over Finite Fields

For an elliptic curve over a finite field, $E/\mathbb{F}_q$, the number of $\mathbb{F}_q$-rational points on E is finite and satisfies the Hasse-Weil bound: writing $\#E(\mathbb{F}_q) = q + 1 - t$, we have $|t| \leq 2\sqrt{q}$ [Sil09, Theorem V.1.1]. The quantity t is called the Trace of Frobenius. One can also ask a more direct question: how many points are there on E? Schoof's algorithm [Sch85] counts points on an elliptic curve over a finite field in polynomial time. The idea is, for a set S of small primes whose product is slightly larger than $4\sqrt{q}$, compute the trace of Frobenius t modulo primes $s \in S$ and then recover t through the chinese remainder theorem.

The number of points on a supersingular curve satisfy the following congruence condition [Was08, Proposition 4.3.1]

$$\#E(\mathbb{F}_q) \equiv 1 \mod p. \tag{III.3}$$

When $p \geq 5$ and the supersingular curve is defined over $\mathbb{F}_p$ (or $\mathbb{F}_{p^2}$ respectively) then, by the Hasse-Weil bound, this condition is the same as the trace of Frobenius being $t = 0$ (or $t = 0, \pm p, \pm 2p$ respectively).

III.1.3 Isogenies, Endomorphisms and Isomorphisms

For two elliptic curves E/k and E'/k, an *isogeny* is a map between points on these curves over the algebraic closure, $\phi : E(\bar{k}) \to E'(\bar{k})$, that is given by rational functions and is also a group homomorphism. By rational functions we mean a quotient of polynomials, namely ϕ is given by $\phi(x,y) = (f(x,y), g(x,y))$ where $f(x,y) = p_1(x,y)/q_1(x,y), g(x,y) = p_2(x,y)/q_2(x,y)$ and p_1, p_2, q_1, q_2 are polynomials with coefficients in $\bar{k}$. In fact, it turns out isogenies can be defined in this simplified form: $\phi(x,y) = (r(x), ys(x))$ where r, s are rational functions [Was08, Section 12.2]. For a field extension K/k, we call the isogeny K-rational if the coefficients of the rational functions lie in K. We say E/k and E'/k are *isogenous* if there exists an isogeny between them. With the exception of the zero map, which maps points on E to the identity on E', every isogeny is surjective [Sil09, Theorem II.2.3]. The set of all elliptic curves that are isogenous to E forms an equivalence class which we call the *isogeny class of E*. We say that an isogeny $\phi : E \to E'$, that map points $(x, y) \in E(k)$ to points $(u(x)/v(x), y \cdot s(x)/t(x)) \in E'(k)$ for polynomials $u, v, s, t \in \overline{k}[x]$, is *separable* if the formal derivative $(u(x)/v(x))'$ is not identically zero and say it is *inseparable* otherwise. Over fields of characteristic zero, every non-zero isogeny is separable. For fields of characteristic $p > 0$, the isogeny can be decomposed into a composition of a separable isogeny with some power of the p-power Frobenius. We summarise this in the following theorem.

Theorem III.2. *[Sil09, Corollary II.2.12] Let E/k and E'/k be elliptic curves defined over a field k of characteristic $p > 0$ and $\phi : E \to E'$ be an isogeny. Then there is a seperable isogeny ϕ_s and an integer $n \geq 0$ such that*

$$\phi = \phi_s \circ \pi^n.$$

Here π is the p-power Frobenius map where $\pi(x, y) = (x^p, y^p)$. In particular we have $\deg(\phi) = p^n \deg(\phi_s)$.

The *degree* of a non-zero isogeny ϕ given in the simplified form with $r(x) = u(x)/v(x)$, denoted $\deg(\phi)$, is equal to $\max\{\deg(u), \deg(v)\}$. The degree of a composition of two isogenies is equal to the product of the degrees of each isogeny [Gal12, Exercise 8.1.12]. For any isogeny ϕ, its kernel is finite and its order is equal to the degree of the separable part of ϕ, namely $\# \ker(\phi) = \deg(\phi_s)$ [Sil09, Theorem III.4.10]. In the special case when the isogeny is separable, the order of the kernel is $\deg(\phi)$. For any isogeny $\phi : E \to E'$, there is a unique isogeny $\hat{\phi} : E' \to E$, called the *dual isogeny*, such that composing these isogenies in either direction gives the multiplication-by-$\deg(\phi)$ map on their respective elliptic curves: $\phi \circ \hat{\phi} = [\deg(\phi)]_{E'}$ and $\hat{\phi} \circ \phi = [\deg(\phi)]_E$ [Sil09, Theorem III.6.1].

An *endomorphism* of E is an isogeny from E to itself. The multiplication-by-m maps are examples of endomorphisms and we call these the trivial endomorphisms. All other endomorphisms are called non-trivial endomorphisms. Denote the set of all endomorphisms including the zero map by $\mathrm{End}(E)$. By equipping $\mathrm{End}(E)$ with the structure of addition and composition, this set forms a ring which we call the *endomorphism ring*. This ring has an embedding of $\mathbb{Z}$ in it as it contains the multiplication-by-m maps. When $\mathrm{End}(E) \neq \mathbb{Z}$, then we say E has *complex multiplication (CM)*. Every elliptic curve over a finite field $\mathbb{F}_q$ has CM since it comes equipt with the *Frobenius map*, $\pi : (x, y) \mapsto (x^q, y^q)$. The trace of Frobenius, t, arises from the following identity which holds for the Frobenius map π: $\pi^2 - [t]\pi + [q] \equiv [0]$ [Was08, Theorem 4.10]. If E has CM then either it is an order in an imaginary quadratic field (a rank two $\mathbb{Z}$-module) or it is a maximal order in a quaternion algebra ramified at the characteristic of the field, p, and ∞ (a rank four $\mathbb{Z}$-module) [Sil09, Corollary III.9.4]. When E is defined over a finite field, then E is ordinary if and only if the endomorphism ring has rank two and E is supersingular if and only if the endomorphism ring has rank four [Sil09, Theorem V.3.1]. Moreover, we define the $\mathbb{F}_q$-rational endomorphism ring, denoted by $\mathrm{End}_q(E)$, to be the set of $\mathbb{F}_q$-rational endomorphisms again equipped with the structure of addition and composition. We note that for supersingular elliptic curves over a prime field $\mathbb{F}_p$, the $\mathbb{F}_p$-rational endomorphism ring is a proper subring of the full endomorphism ring and thus is an order in an imaginary quadratic order.

An *isomorphism* is an isogeny of degree 1. We call two elliptic curves *isomorphic* if such an isomorphism exists. Just like for isogenies, we can form the *isomorphism class of E* consisting of curves which are isomorphic to E. Two elliptic curves are isomorphic if and only if they share the same j-invariant [Sil09, Proposition III.1.4(b)]. In other words isomorphism classes of elliptic curves can be labelled by their j-invariant. If an isomorphism is $k(\sqrt{d})$-rational for a squarefree integer d then we call it a quadratic twist. A common example of quadratic twists that are used in cryptography is the following map between the Montgomery curves $dy^2 = x^3 + Ax^2 + x$ and $y^2 = x^3 + Ax^2 + x$: $(x, y) \mapsto (x, \sqrt{d}y)$. An automorphism is an isomorphism that is also an endomorphism. The set of all automorphisms of an elliptic curve naturally forms a group and the possible automorphism groups of an elliptic curve have been classified [Sil09, Theorem III.10.1].

There is a one-to-one correspondence between finite subgroups of an elliptic curve E and separable isogenies that start at E, up to composition of isomorphisms. This fact has become a key component in the applications of isogenies to cryptography that we shall see in later chapters. We give a formal statement of the correspondence.

Theorem III.3. *[Sil09, Theorem III.4.12] Given an elliptic curve E over a field k and a finite subgroup $G \subseteq E(\bar{k})$, then, up to isomorphism, there is a unique elliptic curve E' such that there is an isogeny $\phi : E \to E'$ whose kernel is G. The elliptic curve E' and the isogeny ϕ are defined over a finite extension of the field k. In the setting where we know an explicit description for the kernel of the isogeny, namely* $\ker(\phi) = G$*, we will often write $E' = E/G$. We call the isogeny* cyclic *if its kernel subgroup, G, is cyclic as a group; in other words, there is a point $P \in G$ such that $G = \langle P \rangle$.*

III.2 Isogeny Graphs

Two elliptic curves defined over a finite field $\mathbb{F}_q$ are isogenous if and only if the number of $\mathbb{F}_q$-rational points on these curves are the same [Tat66]. Hence isogeny classes in finite fields are characterised by their trace of Frobenius. In particular, since we have the congruence condition on the number of points on a supersingular elliptic curve, the number of supersingular isogeny classes is far fewer than ordinary isogeny classes. In particular, Mestre [Mes86] showed that, for a fixed characteristic p, all supersingular elliptic curves lie in a single isogeny class.

Definition III.4. *Let ℓ, p be primes with $\ell \neq p$. The ℓ-*isogeny graph *is a graph whose vertex set is the isomorphism classes of $E/\overline{\mathbb{F}_p}$, labelled by their j-invariants, and two vertices are connected by a directed* [2] *edge if there is a separable ℓ-isogeny between these isomorphism classes.*

*For a fixed elliptic curve $E/\overline{\mathbb{F}_p}$, we define the ℓ-*isogeny graph attached to *E to be the subgraph of the ℓ-isogeny graph whose vertex set is the isogeny class of E.*

For a prime $\ell \neq p$, since the ℓ-torsion subgroup of an elliptic curve is isomorphic to a product of cyclic groups, $E[\ell] \cong (\mathbb{Z}/\ell\mathbb{Z}) \times (\mathbb{Z}/\ell\mathbb{Z})$, then the curve E has exactly $\ell + 1$ subgroups of order ℓ. In the ℓ-isogeny graph, this corresponds to $\ell + 1$ outgoing edges in the ℓ-isogeny graph

[2]This graph can almost be viewed as an undirected graph since every ℓ-isogeny has a dual isogeny of degree ℓ. However there are some exceptions (depending on the automorphism structure of certain elliptic curves), see [Gal12, Remark 25.3.2] for details.

at the vertex $j(E)$. Isogeny graphs have two main components: the ordinary component and supersingular component.

When E is ordinary, the ℓ-isogeny graph attached to E is well understood. This is in large part due to Kohel who in his thesis completely classified the behaviour of ordinary isogeny graphs [Koh96]. In particular, these ordinary isogeny graphs have a volcano structure. We refer the reader to [Sut13] for more details of this volcano structure.

When E is supersingular, the ℓ-isogeny graph attached to E has a different structure. There are finitely many isomorphism classes that arise from supersingular elliptic curves and all such isomorphism classes have j-invariants in $\mathbb{F}_{p^2}$ [Sil09, Theorem V.3.1]. More precisely, this finite number of supersingular isomorphism classes is exactly $\lfloor p/12 \rfloor + \epsilon_p$ [Sil09, Theorem V.4.1] where

$$\epsilon_p = \begin{cases} 0 & : \quad p \equiv 1 \mod 12, \\ 1 & : \quad p \equiv 5 \text{ or } 7 \mod 12, \\ 2 & : \quad p \equiv 11 \mod 12. \end{cases}$$

Hence, any supersingular elliptic curve over $\overline{\mathbb{F}_p}$ has a representation defined over $\mathbb{F}_{p^2}$. From now on all supersingular curves in question will be defined over $\mathbb{F}_{p^2}$. In subsequent work the following result was obtained, the first part is attributed to Mestre [Mes86] and Kohel [Koh96], and the second part is attributed to Pizer [Piz90, Piz95].

Theorem III.5. *The supersingular component in the ℓ-isogeny graph is a connected and $(\ell+1)$-regular (multi)-graph. Moreover this graph is also an expander graph and satisfies the so-called Ramanujan property.*

Due to this connected property we will often refer to the supersingular component of the ℓ-isogeny graph as the *supersingular ℓ-isogeny graph*. The Ramanujan property implies a rapid mixing on the graph, in the sense that for a $O(\log(p))$ length random walk in this graph the induced distribution on the vertices is statistically close to the stationary distribution. In addition this rapid mixing is optimal among all expander graphs in the sense that the length of the walk necessary to approach this stationary distribution is as small as possible.

III.3 The Class Group and the CM Action

Throughout this section K is a number field, $\mathcal{O}$ is an order in K and $\mathfrak{a}$ is an ideal in $\mathcal{O}$. The norm of the ideal $\mathfrak{a}$, denoted by $N(\mathfrak{a})$, is equal to $N(\mathfrak{a}) := \gcd(\{N(\alpha) : \alpha \in \mathfrak{a}\})$. A *fractional ideal* of $\mathcal{O}$ is an $\mathcal{O}$-submodule of K that is of the form $\mathfrak{a} := \frac{1}{d}\mathfrak{b}$ for $d \in K^* := K \setminus \{0\}$ and an $\mathcal{O}$-ideal $\mathfrak{b}$. We say that a fractional $\mathcal{O}$-ideal, $\mathfrak{a}$, is invertible if there is a fractional $\mathcal{O}$-ideal $\mathfrak{a}'$ such that $\mathfrak{a}\mathfrak{a}' = \mathcal{O}$ and denote this ideal by $\mathfrak{a}^{-1} := \mathfrak{a}'$ when it exists. We note that any *principal ideal, $\alpha\mathcal{O}$* for $\alpha \in K^*$, is an invertible fractional ideal since α is non-zero.

Let $I(\mathcal{O})$ and $P(\mathcal{O})$ (respectively) denote the set of invertible fractional $\mathcal{O}$-ideals and principal $\mathcal{O}$-ideals (respectively). As mentioned previously, $P(\mathcal{O})$ is contained in $I(\mathcal{O})$. Equipping $I(\mathcal{O})$ with ideal multiplication, we can turn it into an abelian group with $P(\mathcal{O})$ being a normal subgroup. Thus we can talk about the quotient of these groups. More precisely, define the *ideal-class group* of $\mathcal{O}$ to be

$$\mathrm{cl}(\mathcal{O}) := I(\mathcal{O})/P(\mathcal{O}).$$

Representatives of a fractional ideal $\mathfrak{a}$ within the ideal-class group are typically denoted with a square bracket by $[\mathfrak{a}]$. Moreover, every ideal class $[\mathfrak{a}] \in \mathrm{cl}(\mathcal{O})$ has an integral representative. By this we mean there exists an ideal $\mathfrak{a}'$ that is contained in the order $\mathcal{O}$ and $[\mathfrak{a}'] = [\mathfrak{a}]$ as ideal classes.

The CM action. Let E be an elliptic curve defined over $\mathbb{F}_p$ and $\mathrm{End}_p(E)$ be the $\mathbb{F}_p$-rational endomorphism ring of E. This ring is isomorphic to an order $\mathcal{O}$ in an imaginary quadratic field $K = \mathbb{Q}(\sqrt{-d})$ for some positive and squarefree integer d that not only contains the p-power Frobenius endomorphism, π, but also the order $\mathbb{Z}[\pi]$. Additionally, as an order in K it is a suborder of the ring of integers in K. In the case when E is supersingular and $p \geq 5$, the Frobenius endomorphism satisfies $\pi^2 = [-p]$. So its $\mathbb{F}_p$-rational endomorphism ring is an order in $\mathbb{Q}[\sqrt{-p}]$. The class group of such a field depends on the congruence of p modulo 4. When $p = 1 \mod 4$, there is only one ideal within its class group and so we always have $\mathrm{End}_p(E) = \mathbb{Z}[\pi]$. When $p = 3 \mod 4$, there are two distinct representatives of ideals within its class group and so we either have $\mathrm{End}_p(E) = \mathbb{Z}[\pi]$ or $\mathrm{End}_p(E) = \mathbb{Z}[(1+\pi)/2]$.

For an invertible $\mathcal{O}$-ideal $\mathfrak{a}$, in accordance to Theorem III.3 there is an elliptic curve $E/\mathfrak{a}$ along with an isogeny $\phi_{\mathfrak{a}} : E \to E/\mathfrak{a}$ whose degree is $N(\mathfrak{a})$. In accordance to Theorem III.2, let $\phi_{\mathfrak{a}_s}$ be the separable component of $\phi_{\mathfrak{a}}$, so $\phi_{\mathfrak{a}} = \phi_{\mathfrak{a}_s} \circ \pi^n$ for some positive integer n. Then the kernel of the separable isogeny $\phi_{\mathfrak{a}_s}$ is equal to $\cap_{\alpha \in \mathfrak{a}_s} \ker(\alpha)$ where we treat α as an endomorphism in $\mathrm{End}_p(E)$. We remark that as $\mathcal{O}$-ideals, we have $\mathfrak{a} = \mathfrak{a}_s \cdot (\pi^n \mathcal{O})$. Since principal ideals in $\mathcal{O}$ correspond to endomorphisms, then for two ideals $\mathfrak{a}, \mathfrak{b}$ we have $E/\mathfrak{a} = E/\mathfrak{b}$ if and only if they belong in the same ideal class in $\mathrm{cl}(\mathcal{O})$. Additionally, we have the converse and every $\mathbb{F}_p$-rational isogeny $\psi : E \to E'$ between elliptic curves whose $\mathbb{F}_p$-rational endomorphism ring is isomorphic to $\mathcal{O}$ can be constructed in this way. This results in a free and transitive group action which was originally proved by Waterhouse [Wat69] and later corrected in an erratum by Schoof [Sch87]. The precise statement is the following:

Theorem III.6. *Let $\mathcal{O}$ be an order in an imaginary quadratic field such that the set of elliptic curves over a prime field $\mathbb{F}_p$ whose $\mathbb{F}_p$-rational endomorphism ring is non-empty. Denote this set of curves by $\mathcal{E}_p(\mathcal{O}, \pi)$ where π corresponds to the p-power Frobenius endomorphism on such curves. Then the ideal-class group,* $\mathrm{cl}(\mathcal{O})$ *acts freely and transitively on $\mathcal{E}_p(\mathcal{O}, \pi)$ via the map*

$$\begin{array}{rcl} \mathrm{cl}(\mathcal{O}) \times \mathcal{E}_p(\mathcal{O}, \pi)) & \to & \mathcal{E}_p(\mathcal{O}, \pi)) \\ ([\mathrm{a}], E) & \mapsto & \mathrm{a} \star E := E/\mathfrak{a} \end{array}$$

where $\mathfrak{a}$ is chosen as an integral representative within its ideal class.

III.4 Deuring Correspondence

Recall that for a supersingular elliptic curve E over a finite field $\mathbb{F}_q$ of characteristic p, its endomorphism ring, $\mathrm{End}(E)$, has rank four. Hence there are four linearly independent endomorphisms $\psi_1, \psi_2, \psi_3, \psi_4$ such that every endomorphism, ψ, in the endomorphism ring is a $\mathbb{Z}$-linear combination of them namely, $\psi = a\psi_1 + b\psi_2 + c\psi_3 + d\psi_4$ for some $a, b, c, d \in \mathbb{Z}$. Moreover, since supersingular elliptic curves have j-invariants that lie in $\mathbb{F}_{p^2}$, each endomorphism ψ_i has a representation that is $\mathbb{F}_{p^2}$-rational. Typically ψ_1 is written as 1 which is the identity map on

E since the collection of multiplication maps on E forms a suborder of the endomorphism ring. Additionally $a\psi_1$ refers to the composition of ψ_1 with the multiplication-by-a map, $[a]$.

Not only does it have rank 4 but it is isomorphic to a maximal order in a quaternion algebra that is ramified at p and ∞. This means that it lives inside the algebra $\mathbb{Q}[i,j]$, where $i^2 = -\alpha$, $j^2 = -\beta$ and $ij = -ji$ for some positive $\alpha, \beta \in \mathbb{Q}$. We omit the precise definition of ramification but the choice of α and β here depends on this ramification [Piz80].

This association of an endomorphism ring of a supersingular elliptic curve by such a maximal order can be turned into an equivalence of categories. This is commonly known as the *Deuring correspondence* [Deu41]. We omit the precise details of this correspondence but we refer to either Voight's book [Voi21] or Leroux's PhD thesis [Ler22] for a comprehensive and modern background on this correspondence.

Through this correspondence, the supersingular ℓ-isogeny graphs introduced earlier can be reformulated in the setting of quaternion algebras ramified at p and ∞. Now the supersingular j-invariants that make up set of vertices of the graph are replaced by the set of maximal orders in the quaternion algebra up to conjugacy and the ℓ-isogenies that made up the edges of the graph are replaced by connecting ideals of norm ℓ.

III.5 Smooth and Rough Integers

For integers n, B, we say that n is *B-smooth* if for every prime divisor $p \mid n$ we have $p \leq B$. We also say that n is *B-rough* if for every prime divisor $p \mid n$ we have $p \geq B$. We will often abuse this notation by dropping the B and say that n is smooth (resp. rough) when there is either an implicit choice of B or we simply want n to have small (resp. large) factors. When this notation is not abused, we call B the smoothness (resp. roughness) bound of the integer n. These sorts of numbers have been an interest for mathematicians to study but also seen a number of interesting applications.

One natural question to ask is, for some bound B, how many B-smooth (rough) numbers are there up to some threshold bound X. Fortunately this question has been well studied and we give an overview of the results. To do so we define the functions $\Psi(X,B)$ and $\Phi(X,B)$ that count the number of B-smooth and B-rough integers up to a bound X. In other words,

$$\Psi(X,B) = \#\{1 \leq N \leq X : N \text{ is } B\text{-smooth}\},$$
$$\Phi(X,B) = \#\{1 \leq N \leq X : N \text{ is } B\text{-rough}\}.$$

III.5.1 Distribution of Smooth Numbers

The function $\Psi(X,B)$ was shown, by Dickman and independently by de Bruijn, to be related to the now-called *Dickman-rho* function [Dic30, DB66] which we denote by $\rho : \mathbb{R}^+ \mapsto \mathbb{R}^+$. It is a function that is continuous at $u = 1$, differentiable for $u > 1$ and satisfies the following difference differentiable equation

$$u\rho'(u) = -\rho(u-1), \quad (u > 1);$$
$$\rho(u) = 1, \quad (0 \leq u \leq 1).$$

Then, as $X \to \infty$, it is known that

$$\Psi(X,B) \sim \rho(u)X,$$

where $u = \log(X)/\log(B)$. This shows that, as $X \to \infty$, the probability that a number less than X chosen uniformly at random is B-smooth is asymptotically equal to $\rho(u)$. This will be used to calculate smoothness probabilities in subsequent chapters. More precisely, Hildebrand [Hil86] showed that for fixed $\epsilon > 0$ and uniformly in the domain $X \geq 3$ and $X \geq B \geq \exp\{(\log\log X)^{5/3+\epsilon}\}$, we have

$$\Psi(X,B) = \rho(u)X + O\left(\frac{\rho(u)X\log(u+1)}{\log(B)}\right), \tag{III.4}$$

where $u = \log(X)/\log(B)$ (this result can also be found in [Ten15, Chapter III.5, Corollary 9.3]).

Smooth values of polynomials. For a polynomial $f \in \mathbb{Z}[x]$, we say that an integer *n generates a B-smooth value* of f if the evaluation $f(n)$ is B-smooth. In this case we call n a *B-smooth value of $f(x)$*. In the specific scenario when $f(x) = x(x+1)$ then we call a B-smooth value of $f(x)$ a *B-smooth twin* or *twin B-smooth integers*. This refers to the fact that the consecutive integers $n, n+1$ are B-smooth. Moreover, for an arbitrary polynomial f, we define the function $\Psi_f(X,B)$ to be the number of B-smooth values of the polynomial f up to a threshold bound X. In other words,

$$\Psi_f(X,B) = \#\{1 \leq N \leq X : N \text{ is a } B\text{-smooth value of } f\}.$$

There has been numerous works that have looked into the quantity $\Psi_f(X,b)$ and heuristically one can argue that the probability of smoothness of $f(n)$ is equal to the product of the probabilities of smoothness of each irreducible factor of f. While this heuristic is proven for a certain ranges of X, B [Mar99, Theorem 1.1], these ranges do not apply for the specific ranges that are of cryptographic interest. This having been said, experimentally these estimates are very close to those of cryptographic interest. So now restate the heuristic as given in [CMN21, Heuristic 1].

Heuristic III.7. *Suppose that a polynomial $f \in \mathbb{Z}[x]$ has distinct irreducible factors over $\mathbb{Z}[x]$ of degrees $d_1, \cdots, d_k \geq 1$ respectively. Then, as $X \to \infty$, we have*

$$\Psi_f(X,B) \sim \rho(d_1 u)\cdots\rho(d_k u)X,$$

where $u = \log(X)/\log(B)$.

III.5.2 Distribution of Rough Numbers

The function $\Phi(X,B)$ was shown, by Buchstab, to be related to the now-called *Buchstab* function [Buc37] which will be denoted by $\omega : \mathbb{R}^+ \mapsto \mathbb{R}^+$. It is the unique continuous solution to following difference differentiable equation

$$(u\omega(u))' = \omega(u-1), \quad (u > 2);$$
$$\omega(u) = \frac{1}{u}, \quad (1 \leq u \leq 2).$$

For convenience, we write $\omega(u) = 0$ for $u < 1$. Then, as $X \to \infty$, it is known that

$$\Phi(X,B) \sim \omega(u)\frac{X}{\log(B)},$$

where $u = \log(X)/\log(B)$. More precisely, we have the following. For fixed $\epsilon > 0$ and uniformly in the domain $X \geq 3$ and $X \geq B \geq \exp\{(\log\log X)^{5/3+\epsilon}\}$, we have

$$\Phi(X,B) = (\omega(u)X - B)\frac{e^{\gamma}}{\zeta(1,B)} + O\left(\frac{\omega(u)X}{\log^2(B)}\right), \tag{III.5}$$

where $u = \log(X)/\log(B)$ and $\zeta(1,y) = \prod_{p\leq y} p/(p-1)$ (an account of this result can be found in [Ten15, Chapter III.6, Corollary 7.5]).

III.5.3 Integers with a Large Smooth Divisor

In the later chapters, we will be interested in estimating the probability that a given integer has a large smooth divisor. To do this we introduce the function $\Theta(X,B,D)$ that counts positive integers $N \leq X$ for which there exists a B-smooth divisor $d \mid N$ with $d > D$. In other words,

$$\Theta(X,B,D) = \#\{1 \leq N \leq X : D < \text{largest } B\text{-smooth divisor of } N\}.$$

This function has been previously studied in the literature, see for example [Ten06, Ten15]. In an effort to combine the asymptotic expansions for $\Psi(X,B)$ and $\Phi(X,B)$ mentioned in Equation III.4 and Equation III.5, Banks and Shparlinski showed the following asymptotic expansion for the function $\Theta(X,B,D)$.

Theorem III.8. *[BS06, Theorem 1] For fixed $\epsilon > 0$ and uniformly in the domain*

$$X \geq 3,\ B \geq \exp\{(\log\log X)^{5/3+\epsilon}\},\ B\log B \leq D \leq X/B,$$

we have

$$\Theta(X,B,D) = \Big(\rho(u) + \mathscr{C}_{\omega,\rho}(u,v)\Big)X - \gamma\mathscr{C}_{\omega,\rho'}(u,v)\frac{X}{\log B} + O(\mathscr{E}(X,B,D)),$$

where $u = (\log X)/(\log B)$, $v = (\log D)/(\log B)$, $\gamma := \lim_{n\to\infty}\left[(\sum_{k=1}^{n} 1/k) - \log(n)\right]$ is the Euler–Mascheroni constant, $\mathscr{C}_{f,g}$ is the partial convolution of f with g (as defined in [BS06, pg. 2]), and

$$\mathscr{E}(X,B,D) = \frac{X}{\log B}\left\{\rho(u-1) + \frac{\rho(v)\log(v+1)}{\log B} + \frac{\rho(v)}{\log(v+1)}\right\}.$$

We use this asymptotic expansion to estimate the value of Θ for X, B, D in the relevant range.

Chapter IV

Isogeny-based Cryptography

In this chapter we start to introduce the cryptographic applications that arise from isogenies between elliptic curves. In order to do this we start by giving a description of how to compute isogenies from its kernel in correspondence to Theorem III.3. This will serve as a foundation to building cryptosystems which will be presented later in this chapter. Additionally, we formulate the necessary computationally hard problems in order to guarantee the security of these cryptosystems.

In terms of the concrete cryptosystems, in this chapter we present some of the most well-known cryptosystems. These include the CGL hash function [CLG09], the SIDH key exchange protocol [JDF11, DFJP14], the CSIDH key exchange protocol [CLM⁺18] and also the SQISign signature scheme [DFKL⁺20]. As well as giving a concrete description of each of these protocols, we will give an up to date account of their security as well as their developments over the last few years.

IV.1 Algorithmic Tools for Isogenies

If we wish to build cryptosystems based on computing isogenies then we would hope to be able to have efficient algorithms for computing isogenies. Unfortunately, there is currently no one size fits all approach to compute isogenies efficiently. In some circumstances, we do have efficient algorithms to achieve this. In addition, there are other constructive algorithms that are needed in the construction of cryptographic primitives. These will also be highlighted.

IV.1.1 Computing Isogenies

Historically, the first algorithm presented to compute isogenies is due to Vélu [Vél71]. We present an overview of the algorithm in the general Weierstrass form:

Theorem IV.1. *Let $E/k : y^2 + a_1xy + a_3 = x^3 + a_2x^2 + a_4x + a_6$ be an elliptic curve and let $G \subseteq E(\overline{k})$ be a finite subgroup of points on E. Decompose the subgroup G as a disjoint union of the form*

$$G = \{\infty\} \cup G_2 \cup G^+ \cup G^-$$

where G_2 is the set of 2-torsion points in G and G^+, G^- contain non 2-torsion points of G such that if $P \in G^+$ then $P \notin G^-$ and $-P \in G^-$ (and vice-versa). For each point $Q \in G$ (except $Q = \infty$) set $g_Q^x := 3x_Q^2 + 2a_2x_Q + a_4 - a_1y_Q$, $g_Q^y := -2y_Q - a_1x_Q - a_3$, $u_Q := (g_Q^y)^2$ and if $Q \in G_2$ set $v_Q := g_Q^x$ and otherwise set $v_Q := 2g_Q^x - a_1g_Q^y$. Furthermore, set $S = G^+ \cup G_2$, $v := \sum_{Q\in S} v_Q$ and $w := \sum_{Q\in S}(u_Q + x_Qv_Q)$. Then there is an elliptic curve of the form

$$E' : y^2 + a_1xy + a_3 = x^3 + a_2x^2 + (a_4 - 5v)x + (a_6 - (a_1^2 + 4a_2)v - 7w)$$

such that there is an map $\phi : E \to E'$ that maps points $(x, y) \mapsto (X, Y)$ where

$$X = x + \sum_{Q\in S}\left(\frac{v_Q}{x - x_Q} + \frac{u_Q}{(x - x_Q)^2}\right),$$
$$Y = y - \sum_{Q\in S}\left(u_Q\frac{2y + a_1x + a_3}{(x - x_Q)^3} + v_Q\frac{a_1(x - x_Q) + y - y_Q}{(x - x_Q)^2} + \frac{a_1u_Q - g_Q^xg_Q^y}{(x - x_Q)^2}\right),$$

which defines an isogeny between E and E' whose kernel is the finite subgroup G.

As remarked in Theorem III.3, this isogeny is uniquely determined up to composition of isomorphisms. We refer to [Was08, Section 12.3] for more details towards the proof. Following this, these formulas have been adapted by Elkies [E$^+$98] and Kohel [Koh96] which show that an algorithm for computing isogenies can be done as long as you know the *kernel polynomial*, namely the polynomial $\prod_{Q\in S}(x - x_Q)$.

The presented formulas are based on the general Weierstrass form of an elliptic curve. There are other similar formulas when we are working with a different model for the elliptic curve. For instance, Costello and Husil [CH17] presented formulas in the Montgomery model of an elliptic curve which has been widely adopted in the implementation of many isogeny-based protocols.

The complexity of computing the rational functions of Vélu's formula is determined by the size of the subgroup G. If $N = \#G$, then this can be done with a complexity of $\tilde{O}(N)$. Moreover, for simply evaluating an isogeny at a point, Vélu's formula can evaluate it in linear time $O(N)$. For a full justification of this we refer to [Gal12][Exercise 25.1.14]. This complexity is exponential in the bit size of N. This means that in general this algorithm is not very practical especially if we wish to use it for cryptographic purposes. However, in some circumstances this can be made practical. We illustrate this using the following theorem for which the proof can be found in [Gal12, Theorem 25.1.2].

Theorem IV.2. *Let E, E' be elliptic curves over a field k and let $\phi : E \to E'$ be a separable k-rational isogeny. Then we can write*

$$\phi = \phi_1 \circ \cdots \circ \phi_n \circ [m]$$

where $\phi_1, \cdots, \phi_n$ are prime degree k-rational isogenies and $\deg(\phi) = m^2\prod_{i=1}^n \deg(\phi_i)$.

So isogenies have a decomposition property. This can be exploited to efficiently compute isogenies whose degree is large and smooth. The idea being that instead of using Vélu all in one go to compute the isogeny, one can compute a chain of small degree isogenies and then compose them together to get the intended isogeny. For instance we can compute an isogeny whose degree is a power of two, 2^a, as a composition of 2-isogenies. In this specific instance, we get polynomial-time algorithm for computing isogenies which is very efficient in practice.

We remark that, in accordance with the notation given in Theorem IV.2, the isogeny ϕ is cyclic if and only if for any decomposition of ϕ of the type in the Theorem, we have $m = 1$.

Computing a chain of isogenies using division polynomials. Let ℓ be a small prime. In order to use Vélu's formula to compute an ℓ-isogeny, we need an ℓ-torsion point. If we wish to compute a chain of ℓ-isogenies, then we need a method for generating such torsion points within each step of the isogeny chain. Recall that a point P is an ℓ-torsion point of a curve E if and only if $\psi_\ell(P) = 0$ where ψ_ℓ is the ℓ^{th}-division polynomial as defined in Equation III.2. Also recall that for odd ℓ, by reducing the division polynomial modulo the defining curve equation, we can turn this into a univariate polynomial in x. Therefore we can compute the roots of the division polynomial to give us appropriate torsion points which can be used to compute isogenies. More precisely, to get the next curve in a chain of ℓ-isogenies for odd ℓ, we first compute the irreducible factors (of degree up to $(\ell-1)/2$) of the ℓ-division polynomial. As just mentioned, roots of these factors correspond to the x-coordinate of points of order ℓ [Gal12, Section 25.2]. Choosing one of these points gets you the next step in the chain.

When $\ell = 2$ and the defining equation of E is a general Weierstrass form, then finding 2-torsion points amounts to solving a system of two equations. The first of these equations is $\psi_2(x, y) = 2y + a_1x + a_3 = 0$ and the second is the equation that defines elliptic curve itself. When the short Weierstrass model is used, this translates to computing the roots of the defining polynomial. This strategy also works in the setting when either ℓ is large or ℓ is composite but computing the ℓ-division polynomial is more expensive so might not be feasible in certain circumstances.

Computing a chain of isogenies from an explicit kernel subgroup. Suppose we are given the description of a cyclic subgroup $G = \langle S \rangle \in E(\mathbb{F}_q)$. We wish to compute an isogeny from E whose kernel is G. Again for simplicity, let us assume that the order of G is a prime power ℓ^e. As mentioned before, if ℓ is small then we can efficiently compute this isogeny as a composition of ℓ-isogenies. Additionally, one can make use of the explicit description of G to compute this chain of isogenies.

The point S has order ℓ^e, so the subgroup $\ell^{e-1}S$ has order ℓ. Hence this point will be the kernel of the first isogeny in the chain. When iteratively defining the points $S_0 = S$ and $S_k = \phi_{k-1}(S_{k-1})$ where ϕ_{k-1} is the k^{th} isogeny in the chain, the point $\ell^{e-i-1}S_k$ will give the kernel of each ℓ-isogeny in the chain. Thus the isogeny whose kernel is G is the composition of these ℓ-isogenies.

Vélusqrt. As it shall be noted later, there are some applications for which we have no choice but to evaluate isogenies of relatively large prime degree. So we cannot exploit the decomposition property to compute the isogeny faster. In 2020, Bernstein et. al. [BFLS20] presented the first significant improvement to Vélu's formula by showcasing an algorithm to evaluate an isogeny at a point, P, with a complexity[3] of $\tilde{O}(\sqrt{N})$ if N is the size of the kernel subgroup. The idea behind this speed-up is based on a baby-step giant-step approach to compute the kernel polynomial $\prod_Q(x_P - x_Q)$. Instead of computing this sequentially for all points in the subgroup, one computes

[3]The $\tilde{O}$ accounts for some logarithmic factors in the complexity.

two products whose size is roughly $\sqrt{N}$ and then uses the theory of resultants to recover this evaluation. We omit the precise details of the algorithm, but we remark that the algorithm is similar to the deterministic factoring algorithm due to Strassen [Str76]. Experimentally, they showed that for parameters of cryptographic size, their algorithm could perform better than their Vélu counterparts when N is prime that is not too small (say $N > 150$).

IV.1.2 Constructive Deuring Correspondence

As mentioned in Section III.4, there is a correspondence between supersingular elliptic curves and maximal orders in a quaternion algebra through the endomorphism ring of the elliptic curve. In this section we explore the following problem known as the *constructive Deuring correspondence*: For a fixed maximal order $\mathcal{O}$ in the quaternion algebra ramified at p and ∞, find a supersingular elliptic curve whose endomorphism ring is isomorphic to $\mathcal{O}$. This is a problem that has seen constructive applications, so being able to do this efficiently is necessary for these applications.

In order to do this in polynomial-time, one needs the following algorithm due to Kohel, Lauter, Petit and Tignol (often abbreviated as KLPT) [KLPT14]. This algorithm aims to solve the quaternion isogeny path problem, namely given two maximal orders find a path in the quaternion ℓ-isogeny graph or equivalently a connecting ideal of norm ℓ^e. We omit the precise details of this algorithm and refer the reader to [KLPT14] for these details.

Solving the constructive Deuring correspondence can be done as follows (as described in [EHL+18]). Suppose that we have a maximal order $\mathcal{O}$ within the quaternion algebra. Our goal is to find a supersingular elliptic curve such that $\mathrm{End}(E) \cong \mathcal{O}$. In order to do this, first construct a supersingular elliptic curve, E_0, whose endomorphism ring can be efficiently computed and is isomorphic to a maximal order $\mathcal{O}_0$. For instance, when $p = 3 \mod 4$, the elliptic curve of j-invariant 1728, $E_0 : y^2 = x^3 + x$, is supersingular and has an endomorphism ring equal to

$$\mathrm{End}(E) = \langle 1, \iota, \frac{\iota + \pi}{2}, \frac{1 + \iota \circ \pi}{2} \rangle,$$

where π is the p-power Frobenius and $\iota(x, y) = (-x, iy)$ with $i^2 = -1$. Outside of this setting there are other such supersingular elliptic curves [Brö09]. Once one has this curve, one uses the KLPT algorithm applied to the maximal orders $\mathcal{O}_0$ and $\mathcal{O}$ to find a connecting ideal I between them. We then define the kernel subgroup of E_0 associated to I, namely the subgroup $E[I] = \cap_{\alpha \in I} \ker(\alpha) \subseteq E_0(\overline{\mathbb{F}_p})$ and use the isogeny computation tools mentioned previously to compute an isogeny $\phi_I : E_0 \to E_0/E[I]$. Then the elliptic curve $E := E_0/E[I]$ has an endomorphism ring isomorphic to $\mathcal{O}$ and thus solves the constructive Deuring correspondence.

This original formulation of the solution to the constructive Deuring correspondence by Eisentraeger et al. [EHL+18] is only a heuristic polynomial-time solution. In 2021, Wesolowski [Wes22] showed that one can get a provable polynomial-time solution assuming the generalised Riemann Hypothesis (GRH). We remark that while for practical applications one would use specially chosen primes to suit the application, implementations of this algorithm has been done in general characteristic [Ray18, KYN+21, EPSV23]

IV.2 Computationally Hard Isogeny Problems

In the previous section we focused our attention on computing isogenies directly from a kernel subgroup. In order to build cryptosystems based on these isogenies one would hope that the converse problem is intractable – namely, given two supersingular elliptic curves find an isogeny between them. Here we discuss these hard problems and the known general strategies for solving them. We focus our attention on the supersingular variants of these problems. We refer the reader to other sources for the ordinary analogue of these problems [Koh96, Gal99, BS11, CJS14, Rob22].

A priori, all elliptic curves should be defined over $\overline{\mathbb{F}_p}$. However, since we are interested in supersingular elliptic curves and every such curve has a representation that is defined over $\mathbb{F}_{p^2}$, we can assume that all supersingular elliptic curves discussed here are defined over $\mathbb{F}_{p^2}$. Occasionally supersingular elliptic curves will be defined over the algebraic closure purely for the sake of generality.

IV.2.1 Hard Problems for Security

Fundamental to all of isogeny-based cryptography are the following isogeny problems.

Problem 1 (Supersingular isogeny problem). *Given two supersingular elliptic curves $E/\overline{\mathbb{F}_p}$ and $E'/\overline{\mathbb{F}_p}$, find an isogeny $\phi : E \to E'$.*

Problem 2 (Supersingular ℓ-isogeny path problem). *Given two supersingular elliptic curves $E/\overline{\mathbb{F}_p}$ and $E'/\overline{\mathbb{F}_p}$, find a path in the ℓ-isogeny graph that connects $j(E)$ and $j(E')$.*

Both problems are tasked to find an isogeny between two elliptic curves that lie in an isogeny class. We note that, by Tate's theorem, determining whether two elliptic curves lie in the same isogeny class can be translated to counting points on the respective elliptic curves and, as mentioned in Section III.1.2, can be done in polynomial-time. The main difference between the two problems is that the later problem asks one to find a specific type of isogeny that has a certain degree. The first attempt to solve these problems is due to Galbraith [Gal99].

The best known algorithms for solving these problems have an exponential classical and quantum complexity. Concretely, the Delfs-Galbraith algorithm [DG16] solves the general problem with a complexity $\tilde{O}(\sqrt{p})$. The idea behind the algorithm is starting from each the curve E and E' go on a walk until you reach subfield curves E_0 and E_0' that are defined over the prime field $\mathbb{F}_p$. Solving the isogeny problem from E_0 to E_0' is now easier and can be done in $O(p^{1/4})$ [DG16, Algorithm 1] – from which you solve the isogeny problem from E to E'. There are around $O(\sqrt{p})$ subfield vertices in the isogeny graph [Cox89] among the $p/12$ total vertices. So on average one would expect to have to go on a walk of length $O(\sqrt{p})$ before one finds these subfield curves – thus yielding the $O(\sqrt{p})$ complexity for the algorithm. Recently, Santos, Costello and Shi [CRSCS22] provided a variant of the Delfs-Galbraith algorithm which achieves the same complexity but incorporates some speed-ups in the process of finding the subfield curves. So in practice it performs better than the original algorithm. The best quantum algorithm to solve the isogeny problem is due to Biasse, Jao and Sankar [BJS14] and achieves a complexity of $\tilde{O}(p^{1/4})$.

The algorithms described above do not assume any structure on the underlying isogeny and just ask us to find an isogeny between the curves. If the degree of the isogeny is known and is not too large in comparison to an average isogeny, then a meet-in-the-middle strategy could solve the isogeny problem faster. The idea is that you go on multiple walks whose length is around the square root of its degree from both sides and store all elliptic curves that result from the walk. Since the degree is small and there is a known isogeny between the curves, then there should be a collision between these sets – thus giving an isogeny between the curves. Unlike the Delfs-Galbraith algorithm, this is not memory free and requires roughly the same amount of storage compared to its runtime. The van-Oorschot Wiener (vOW) algorithm [VOW99] is a variant of the meet-in-the-middle algorithm which requires less storage at a cost of a slightly lower complexity. In practice for cryptographic sized parameters, the van-Oorschot Wiener algorithm performs better over its meet-in-the-middle counterpart [ACVCD+19].

Next we look at computing endomorphisms of a supersingular curve.

Problem 3 (Supersingular endomorphism ring problem). *Given a supersingular elliptic curve $E/\overline{\mathbb{F}_p}$, compute its endomorphism ring* $\mathrm{End}(E)$. *Equivalently, find four endomorphisms of E that forms a basis for the ring.*

Problem 4 (One supersingular endomorphism problem). *Given a supersingular elliptic curve $E/\overline{\mathbb{F}_p}$, compute a non-trivial and cyclic endomorphism of E whose degree is either smooth or some prime power.*

These two computational problems date back to the thesis Kohel [Koh96], who gave an algorithm for computing a rank 4 suborder of the endomorphism ring that probabilistically terminates with an expected $\tilde{O}(p)$ operations in the underlying finite field. The idea behind the algorithm is to go on walks in a supersingular isogeny graph until cycles in the graph are found. These cycles constitute endomorphisms of the starting curve. Thus one can view the one supersingular endomorphism problem as a special case of the supersingular endomorphism ring problem. The current state-of-the-art when it comes to computing endomorphism rings is due to Eisentrager et al. [EHL+20]. Their probabilistic algorithm runs in $\tilde{O}(p^{1/2})$ time and follows a similar approach undertaken by Kohel.

Remark IV.3. *There is a relationship between the isogeny problems and endomorphism ring problems. As just described, an isogeny graph is used to compute endomorphisms. Conversely, the KLPT algorithm gives a method to find isogenies between supersingular elliptic curves given their endomorphism rings. Thus these problems are equivalent. This equivalence was heuristically proved by a series of authors [PL17, EHM17, EHL+18], but recently Wesolowski [Wes22] proved this equivalence assuming the GRH.*

IV.2.2 Hard Problems without a trusted setup

In recent years, a number of protocols require starting with a supersingular elliptic curve such that no one (not even the individual who finds the supersingular curve) knows its endomorphism ring [DFMPS19, BDF21]. This includes the protocol described in Chapter V of this thesis. We call such a supersingular elliptic curve a *hard supersingular curve*. This task is very much related to finding a random supersingular j-invariant since for such a j-invariant the endomorphism ring of the corresponding elliptic curve should be unknown and infeasible to compute.

Currently there are no known solutions to construct such a hard supersingular curve that do not require a trusted setup. The idea to do this with a trusted setup is to have multiple parties work together to compute an isogeny composition of large degree. Each party will know some part of the isogeny chain but recovering the whole isogeny chain would require solving an isogeny problem. This was recently formalised by Basso et al. [BCC+23]. It is worth noting that some attempts have been made [BBD+22, MMP22] to find these hard curves without a trusted setup. However each method is shown to reveal some information about its endomorphism ring and so cannot be used as hard curves.

IV.3 CGL Hash Function

Proposed in 2006 by Charles, Goren and Lauter [CLG09], the CGL hash function was the first protocol designed using supersingular isogenies. The idea behind this hash function is to start at a known supersingular elliptic curve and go on a non-backtracking walk on a supersingular isogeny graph determined by the functions input. The resulting vertex/j-invariant at the end of this walk would be the hash of the input. It exploits the Ramanujan property of these supersingular isogeny graphs to ensure the output of the hash for small enough inputs appears to be chosen uniformly at random among all supersingular j-invariants. We give a more detailed description of this hash function as originally formulated by Charles, Goren and Lauter as well as state some recent developments.

IV.3.1 Protocol Description

Let $E_0/\mathbb{F}_{p^2}$ be a known supersingular elliptic curve given by a short Wierstrass equation, $E_0 : y^2 = x^3 + A_0x + B_0$, and $m = [m_0, \cdots, m_{k-1}] \in (\mathbb{Z}/2\mathbb{Z})^k$ be a length k binary string with $k = O(\log(p))$. The 2-division polynomial of E_0 is $2y$ and hence the 2-torsion points on this curve have x-coordinates that are roots of the defining polynomial which we shall call $x_{0,0}, x_{0,1}, x_{0,2}$. Based on the bit value m_0 we choose one of these roots and compute a 2-isogeny, ϕ_0, whose kernel is generated by the corresponding 2-torsion point. This gives us a supersingular curve E_1 defined by an equation $y^2 = x^3 + A_1x + B_1 = (x - x_{1,0})(x - x_{1,1})(x - x_{1,2})$ which is 2-isogenous to E_0. Before we do the next step we make a note of which of these new roots takes us back to E_0, this constitutes as backtracking in the graph which we want to avoid. Recall that this amounts to computing the kernel of the dual isogeny $\ker(\hat{\phi}_0)$ and noting which of these roots generates this kernel. Without loss of generality, we assume this root is $x_{1,2}$. For the next step we use the bit m_1 to choose one of the other two roots, $x_{1,0}, x_{1,1}$, and again compute a 2-isogeny, ϕ_1, whose kernel is generated by the corresponding 2-torsion point. We repeat this for all bits in the bit string m and end up with a path from E_0 to a supersingular elliptic curve E_k. This final curve (or it's j-invariant $j(E_k)$) will be the hash of the message m. It will be convenient to give some notation to this. We shall call $\Phi_2(E_0, m) := E_k$. More generally, for a small prime ℓ we denote by $\Phi_\ell(E, m)$ the elliptic curve obtained by going on a walk in the ℓ-isogeny graph attached to E starting at E and dictated by an input $m \in \mathbb{Z}/(\ell + 1)\mathbb{Z} \times (\mathbb{Z}/\ell\mathbb{Z})^{k-1}$.

A variant of this hash function was proposed by Doliskani, Pereira and Barreto [DPB17] in which they exploit choosing primes of the form $p = 2^n f - 1$ and doing isogeny computations from a kernel subgroup. They show that their hash function is significantly faster than the original CGL hash function. Additionally, a generalisation of this function was presented as part of the work by De Feo et al. [DFDdSGF$^+$21].

Remark IV.4. *The description given above is specific to the case of using supersingular isogeny graphs. The original article phrased a more general framework by constructing hash function from expander graphs and Ramanujan graphs. These include the Lubotzky-Phillips-Sarnak (LPS) expander graphs. The idea is the same, the only difference is that you work on a different graph. The hash function based on the LPS graphs has been extensively cryptanalysed [PLQ08] which conclude that computing preimages of such a graph is not as difficult as first imagined.*

IV.3.2 Cryptanalysis

The hash function described above was proved to be preimage resistant and collision resistant against the problems defined in the previous section. More specifically, if the isogeny path problem is computationally hard then the function is preimage resistant and if the one endomorphism problem is computationally hard then the function is collision resistant. With regards to the one endomorphism problem, it specifically targets finding an endomorphism of degree ℓ^{2k} since that constitutes a cycle in the isogeny graph.

In some sense, the knowledge of the endomorphism ring of the starting elliptic curve could reveal certain endomorphisms that could counter the collision resistance claim mentioned above. More concretely, with aid of the constructive Deuring correspondence, work by Eisentraeger et. al. [EHL$^+$18] showed that the preimage and collision resistance claims are equivalent to the endomorphism ring problem. Hence the hash function is not secure if one knows the endomorphism ring of the starting elliptic curve.

When the starting curve is a hard curve, namely when no one knows its endomorphism ring including the person who found the curve, then the hash function remains secure and there are no known ideas to break this function in this setting. As mentioned in Section IV.2.2, current known techniques for achieving this require a trusted setup. This is not ideal but it is the only solution to accomplish this.

IV.4 SIDH/B-SIDH

Supersingular Isogeny Diffie Hellman (or SIDH) is a key exchange protocol proposed by Jao and De Feo [JDF11, DFJP14]. The idea is that in order for Alice and Bob to establish a secret shared key, Alice performs computations solely within one supersingular ℓ-isogeny graph and Bob will do computations in another supersingular ℓ-isogeny graph. The exchange mimics the traditional Diffie-Hellman protocol in which each party performs some computation, exchanges some information and repeats their computation with the new information. In order for the scheme to commute and both Alice and Bob to obtain a shared secret, some additional information needs to be revealed. As shall be noted later, this additional information has proved fatal to the security of the scheme. Figure 5 summaries the key exchange.

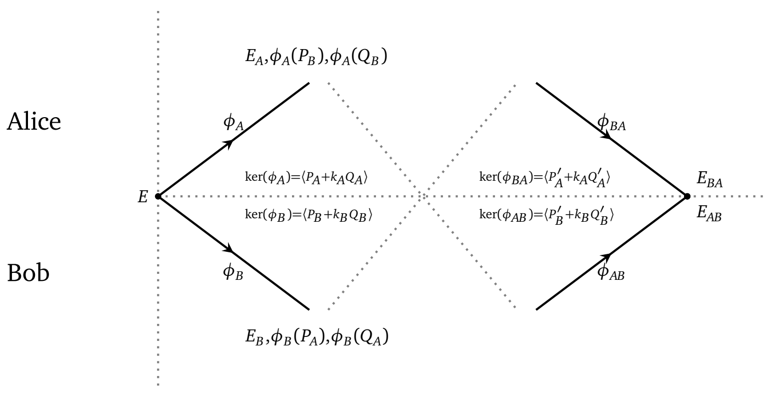

$E_A, \phi_A(P_B), \phi_A(Q_B)$
Alice
ϕ_A
ϕ_{BA}
$\ker(\phi_A) = \langle P_A + k_A Q_A \rangle$
$\ker(\phi_{BA}) = \langle P'_A + k_A Q'_A \rangle$
E
E_{BA}
$\ker(\phi_B) = \langle P_B + k_B Q_B \rangle$
$\ker(\phi_{AB}) = \langle P'_B + k_B Q'_B \rangle$
E_{AB}
Bob
ϕ_B
ϕ_{AB}
$E_B, \phi_B(P_A), \phi_B(Q_A)$

Figure 5: Supersingular isogeny Diffie-Hellman (SIDH) key exchange protocol.

IV.4.1 Protocol Description

Let ℓ_A, ℓ_B be distinct small primes (typically 2 or 3) and let $p := \ell_A^a \cdot \ell_B^b \pm 1$ be a prime of cryptographic size for which $\ell_A^a \approx \ell_B^b$. Construct a supersingular elliptic curve $E/\mathbb{F}_{p^2}$ [Brö09] that has $(p \mp 1)^2$ points (when the trace of Frobenius is $\pm 2p$), namely we have $\#E(\mathbb{F}_{p^2}) = \left(\ell_A^a \ell_B^b\right)^2$. This makes the ℓ_A^a and ℓ_B^b-torision subgroups of E entirely contained within $\mathbb{F}_{p^2}$-rational points of E. Let $P_A, Q_A, P_B, Q_B \in E(\mathbb{F}_{p^2})$ be points on E that generate their respective torsion subgroups: $\langle P_A, Q_A \rangle = E[\ell_A^a]$, $\langle P_B, Q_B \rangle = E[\ell_B^b]$.

The key exchange protocol works as follows. Alice chooses a secret scalar $k_A \in \mathbb{Z}/\ell_A^a\mathbb{Z}$ and computes the point $S_A = P_A + k_A Q_A$. Then using Problem 1 from the previous section, she computes an isogenous elliptic curve, E_A, along with an isogeny $\phi_A : E \to E_A$ whose kernel is generated by the point S_A. Alice will also compute the images of the points P_B, Q_B under this isogeny, $\phi_A(P_B), \phi_A(Q_B)$. Whilst this is happening, Bob will similarly choose a secret scalar $k_B \in \mathbb{Z}/\ell_B^b\mathbb{Z}$ and compute the point $S_B = P_B + k_B Q_B$. Then he computes an isogenous elliptic curve, E_B, along with an isogeny $\phi_B : E \to E_B$ whose kernel is generated by the point S_B as well as the images of P_A, Q_A under this isogeny, $\phi_B(P_A), \phi_B(Q_A)$. Alice (and Bob) store the triple $[E_A, \phi_A(P_B), \phi_A(Q_B)]$ (and $[E_B, \phi_B(P_A), \phi_B(Q_A)]$) as their public keys and they exchange these keys with one another. Alice now computes the point $S'_A = \phi_B(P_A) + k_A \phi_B(Q_A)$ and an isogeny $\phi_{BA} : E_B \to E_{BA}$ whose kernel is generated by S'_A. Similarly, Bob computes the point $S'_B = \phi_A(P_B) + k_B \phi_A(Q_B)$ and an isogeny $\phi_{AB} : E_A \to E_{AB}$ whose kernel is generated by S'_B. After all of this, Alice and Bob arrive at the curves E_{BA} and E_{AB} which have the same j-invariant which they use as their shared secret key. These j-invariants are the same since the composition of the isogenies $\phi_{BA} \circ \phi_B$ and $\phi_{AB} \circ \phi_A$ have the same kernel.

The idea behind this key exchange can be turned into an ElGamal style public key encryption scheme when working in the random oracle model. In order to encrypt a message $m \in \{0,1\}^\lambda$ on a public key $[E_A, \phi_A(P_B), \phi_A(Q_B)]$, one computes $\mathscr{H}(j(E_{AB}))$, where $\mathscr{H}$ is a cryptographic hash function that is modelled as a random oracle, and one produces a ciphertext $c = (E_B, \phi_B(P_A), \phi_B(Q_A), m \oplus \mathscr{H}(j(E_{BA})))$. Here $\oplus$ is the standard XOR operation on binary strings. To decrypt this ciphertext c, one uses the SIDH triple $(E_B, \phi_B(P_A), \phi_B(Q_A))$ and computes $\mathscr{H}(j(E_{BA}))$. Since this is the same as $\mathscr{H}(j(E_{AB}))$, then the message can be recovered as $m = c \oplus \mathscr{H}(j(E_{BA}))$.

Exploiting the twisted torsion. In the SIDH protocol, there is flexibility in choosing how many points we require our supersingular elliptic curves to have and work with this throughout the scheme. One chooses a prime of a particular form, say $p = \ell_A^a \cdot \ell_B^b - 1$, as well as fix a quadratic twist, namely supersingular curves with $p+1$ points, and work solely with those curves. One could loosen this and attempt to construct something that uses both quadratic twists. This was initially exploited by Costello [Cos20] who designed the key agreement protocol B-SIDH as an adaptation of SIDH. The idea here is for Alice to do isogeny computations solely within the $p+1$ and Bob to do isogeny computations solely within the $p-1$ torsion and follow the same design as has been done in SIDH. A priori, this would require working over $\mathbb{F}_{p^4}$ in order to be able to work with the appropriate quadratic twist. However if you work in the Montgomery model and only do x-only arithmetic then you can do everything within $\mathbb{F}_{p^2}$. The main advantage here is that you can make the size of the prime a lot smaller in comparison to standalone SIDH. This is because each party does overall more computation – here an isogeny computation of size p is done versus $\sqrt{p}$ as in SIDH. This gives you much smaller keys over SIDH. The tricky aspect of this is finding suitable primes p such that both $p+1$ and $p-1$ are smooth. Unlike in SIDH, we can not make this smoothness optimally small. According to the current literature, the smallest smoothness bound of p^2-1 for a prime p of around 240-256-bits that has been found is around 2^{15} [CMN21, Appendix A]. This large smoothness bound impacts the performance of the scheme and therefore makes it a lot slower. Since this idea came to light, the topic of finding primes such that $p \pm 1$ are both smooth (or at least contain a large smooth cofactor) has been looked into by others [DFKL+20, CMN21, DFDdSGF+21]. This is related to the problem of finding twin-smooth integers. If $(m, m+1)$ is a smooth twin and their sum $p = 2m+1$ is prime then p^2-1 is also smooth. This topic will be explored further in Chapter VI.

IV.4.2 Security Analysis and Cryptanalysis

Fundamental to the security of SIDH is the so-called "supersingular isogeny with torsion points" problem. The task here is given $(E, P_B, Q_B, E_A, P'_B, Q'_B)$, to recover the isogeny of degree $\ell_A^{e_A}$ that connects them. This can be thought of as a constrained version of the more general supersingular ℓ-isogeny path problem since we are given more information about the isogenies structure that was previously not given. For a long time it was thought that the meet-in-the-middle approach and the vOW alternative were the best algorithms to solve this isogeny problem. We note that this approach does not use the torsion point images, rather that we know the degree of the isogeny and that it is relatively short in comparison to an expected degree of an isogeny between supersingular elliptic curves.

Since its inception, it had been suspected that the additional torsion point information given in the public key could be exploited to make the isogeny used in SIDH easier to compute than its more general counterpart. This thinking has had its own evolution over the last few years. We give a brief overview of the main points and takeaways.

Adaptive security. From the perspective of the public key encryption scheme based on SIDH, we have a security reduction that shows that the scheme is IND-CPA secure, but there has been no security proof to show that it is IND-CCA secure. The main difference now is that in the IND-CCA game the adversary is given access to a decryption oracle. In 2016, Galbraith

Petit Silva and Ti [GPST16] showed that this would never hold for the SIDH scheme. More precisely, they showed that given access to the oracle $\mathcal{O}$ defined below, there is a polynomial-time algorithm to recover the secret scalar k_A given the public key E_A, P'_B, Q'_B. The oracle takes as input a tuple consisting of a supersingular curve and two torsion points of the correct order on the curve and outputs the shared key that would be obtained using the scalar k_A.

We give an overview of the attack in the setting when $\ell_A = 2$ and refer to [GPST16] for some details when ℓ_A is odd. The idea is to query the oracle with a slightly altered triple to what the public key actually is in such a way that one can obtain a bit of the secret scalar k_A. This can be carried out as follows. Let E_B, P'_A, Q'_A be a public key obtained honestly by the attacker through a secret scalar k_B and $j(E_{BA})$ be the shared key that the attacker would obtain from Alice's public key. Then the oracle is queried on the modified triple $(E_B, P'_A, Q'_A + 2^{a-1}P'_A)$ to obtain a j-invariant j. The inclusion of the additional 2-torsion point here means that $j = j(E_{BA})$ if and only if k_A is even [GPST16, Lemma 3.1]. So from this, a single bit of information about k_A is revealed. One can continue the attack in a similar fashion whereby each query to the oracle reveals another bit of information. We omit these details here but they can be found in [GPST16, Section 3.2].

A number of countermeasures have been proposed in an attempt to prevent the possibility of such an attack. The initial countermeasure presented by [GPST16] is to simply apply a variant of the Fujisaki Okamoto transform [FO99, KLM+15]. The adoption of such a transform was used in the submission of the NIST post quantum standardisation of SIKE (Supersingular Isogeny Key Encapsulation) [JAC+17]. It was presented as an IND-CCA secure KEM. There have been other countermeasures proposed which modify the construction at the isogeny level [AJL18, UJ20, FP21a], however each of these countermeasures have been shown to have an adaptive attack of some description [DGL+20, BKM+20, GL22]. The only countermeasure which allows for SIDH to remain as a non-interactive key exchange requires proving in zero-knowledge that you know the secret kernel [DFDGZ23]. We note that this requires performing multiple rounds of SIDH and thus is a very costly procedure.

Initial torsion point attacks. The oracle needed in the previous attack makes the adversary quite strong. In reality, an attacker may not have access to such an oracle. The first passive attack that explicitly utilised the torsion point information was due to Petit [Pet17] from 2017. His attack does not break SIDH as described above but breaks a unbalanced variant of SIDH. For convenience, denote $M := \ell_A^{e_A}$ and $N := \ell_B^{e_B}$. The idea behind the attack is to use information about the endomorphism ring of the input curve E to find an endomorphism of the output curve E' whose degree is of the form Ne for some smooth and/or small e and to compute it using the torsion information of our isogeny. From this endomorphism, one can recover the secret isogeny. In the case when E is the curve with j-invariant 1728, one requires the following unbalanced parameters $M > p$ and $N > M^4$ to make this attack practical. In 2021, this approach was revisited and improved [dQKL+21]. Instead of finding an endomorphism on E' of degree Ne and recovering the secret isogeny from this endomorphism, they are able to it from an endomorphism of degree N^2e. This means that they can reduce the size of the unbalanced parameters to $p > M$ and $N > pM$. Despite these improvements, they are not applicable to parameters used for SIDH since $M \approx N \approx \sqrt{p}$. Nevertheless, they were able to show that a group key agreement protocol based on SIDH [AJJS19] could be broken with 6 or more parties.

Higher-dimensional isogenies. In 2022, a series of papers forever changed the outlook for SIDH in its current form as a viable scheme for practical use. In the first of these works by Castryck and Decru [CD23], a polynomial-time algorithm was presented that solves the supersingular isogeny with torsion points problem assuming that one knows the endomorphism ring of the starting curve E. In the work by Maino and Martindale [MMP$^+$23] they presented a subexponential time algorithm that solves this problem without the knowledge of the endomorphism ring. Subsequently, Robert [Rob23] presented a polynomial-time algorithm without knowledge of the endomorphism ring.

In all of these algorithms a common theme is used. One takes the isogenies and torsion points and embeds them into an isogeny of higher dimension through a lemma by Kani [Kan97]. Then one solves this corresponding isogeny problem with the help of this lemma. In [CD23,MMP$^+$23] isogenies in dimension 2 are used, while in [Rob23] isogenies in dimension 4 and 8 are used.

The main feature which makes these attacks work is that not only do we know the degree of the isogeny which we want to recover but we also know the torsion point images and exactly how they are related to the torsion points on the base curve. As part of a countermeasure to these attacks proposed by Fouotsa, Moriya and Petit [FMP23], two schemes have been presented. The first one, which they call M-SIDH, masks the torsion point images by multiplying them by a random quadratic square root of 1. The second one, which they call MD-SIDH, hides the degree of the isogeny. Both of these countermeasures require the size of the parameters to be larger than for SIDH been currently which makes the use of these countermeasures in a practical setting unlikely.

Remark IV.5. *The upshot of all of this is that one should be careful revealing torsion point images on a whole basis unless it is masked in some appropriate way. Not only is the security of traditional SIDH broken but also so are number of other protocols that uses the SIDH framework [YAJ$^+$17, GPS17, BKW20, DFDdSGF$^+$21, EJKM22]. We note that a recent protocol based on SIDH has been proposed that incorporates these countermeasures to the SIDH attacks [Bas23].*

IV.5 CSIDH

Commutative Supersingular Isogeny Diffie-Hellman (or CSIDH) is another key exchange protocol which unlike the SIDH counterpart still remains secure. It was proposed by Castryck, Lange, Martindale, Panny and Renes [CLM$^+$18] and utilises the CM-action introduced in Section III.3. It has a bit more history compared to SIDH and the underlying idea goes back to the work of Couveignes [Cou06] and was independently rediscovered by Rostovtsev and Stolbunov [RS06]. In these original constructions they used ordinary curves and were very slow [DFKS18]. The difference in CSIDH is that not only are supersingular curves used but they are also able to pick parameters that make the scheme reasonably efficient. Figure 6 summaries the key exchange.

IV.5.1 Parameter Setup

Let $\ell_1, \cdots, \ell_n$ be small odd primes such that $p = 4\ell_1 \cdots \ell_n - 1$ is a prime of cryptographic size. Let $E/\mathbb{F}_p$ be a supersingular elliptic curve whose $\mathbb{F}_p$-rational endomorphism ring is $\mathbb{Z}[\pi]$. A

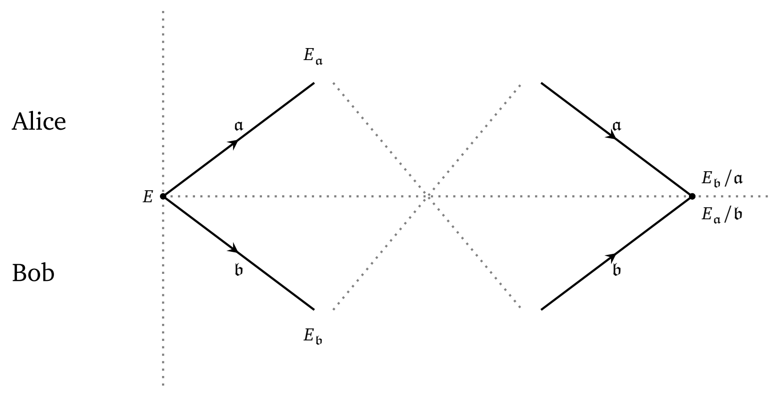


Figure 6: Commutative supersingular isogeny Diffie-Hellman (CSIDH) key exchange protocol.

typical choice is the elliptic curve with j-invariant 1728: $E : y^2 = x^3 + x$.

For such a prime p, the supersingular ℓ_i-isogeny graph attached to E is a disjoint union of cycles [CLM⁺18, Theorem 4]. One such cycle consists of the subset of curves whose $\mathbb{F}_p$-rational endomorphism is $\mathbb{Z}[\pi]$. The ideal class group of $\mathbb{Z}[\pi]$ contains the ideals of the form $\mathfrak{l}_i := (\ell_i, \pi - 1)$ and $\overline{\mathfrak{l}_i} := (\ell_i, \pi + 1)$. We note that the principal ideal generated by ℓ_i splits into these two ideals. In particular, the action of one of these ideals on E constitutes a step in the cycle mentioned above. Moreover, the kernel of the isogeny generated by the ideal $\mathfrak{l}_i$ consists of $\mathbb{F}_p$-rational points of order ℓ_i.

Ideally, for what will be described, one would like to sample elements of the class group uniformly at random. This would require precomputing the class group which as mentioned previously is an expensive computation which has only been done for some parameters [BKV19]. However, one can get around this via some heuristic arguments related to the ideals $\mathfrak{l}_i$. Namely, that the number of pairs of ideals of the form $\mathfrak{l}_1^{e_1}\mathfrak{l}_2^{e_2}\cdots\mathfrak{l}_n^{e_n}$ for small exponents e_i that coincide within the class group is negligible. So one can sample ideal classes by choosing exponents e_i within some range, $e_i \in \{-m_i, \cdots, m_i\}$ for some m_i and compute the ideal $\mathfrak{l}_1^{e_1}\mathfrak{l}_2^{e_2}\cdots\mathfrak{l}_n^{e_n}$. Assuming the heuristic, this distribution of ideals within the class group is statistically close to uniform.

IV.5.2 Protocol Description

The key exchange protocol works as follows. Using the special ideals defined in the previous section, Alice samples an ideal $\mathfrak{a} \in \mathrm{cl}(\mathbb{Z}[\pi])$ and computes the curve $E_\mathfrak{a} := E/\mathfrak{a} = \mathfrak{a} \star E$. Whilst this is happening Bob also samples an ideal $\mathfrak{b} \in \mathrm{cl}(\mathbb{Z}[\pi])$ and computes the curve $E_\mathfrak{b} := E/\mathfrak{b} = \mathfrak{b} \star E$. Alice (and Bob respectively) store $E_\mathfrak{a}$ (and $E_\mathfrak{b}$ respectively) as their public keys and they exchange these public keys. Upon receiving Bob's public key, Alice computes the curve $E_\mathfrak{b}/\mathfrak{a} = \mathfrak{a} \star E_\mathfrak{b}$. Similarly, upon receiving Alice's public key, Bob computes the curve $E_\mathfrak{a}/\mathfrak{b} = \mathfrak{b} \star E_\mathfrak{a}$. Since the group action is free and transitive, the curves $E_\mathfrak{b}/\mathfrak{a}$ and $E_\mathfrak{a}/\mathfrak{b}$ will be the same curve and hence both parties use this as their shared key.

The natural way of turning CSIDH into a public key encryption scheme is through a similar ElGamal technique as presented in the context of SIDH. Once again this requires the use of a hash function and one obtains a ciphertext by xoring a message with the hash of the shared secret. The inclusion of the hash function is something that one would like to avoid. The formalities of the security proof was carried out by Stolbunov [Sto09] in the setting of the ordinary instantiation, but can be easily adapted to the supersingular setting.

This was remedied by [MOT20] in which an IND-CPA secure public key encryption scheme called SiGamal is constructed. An IND-CCA version of SiGamal was presented in [FP21b] which is called SimS. In these protocols, the use of hash functions is replaced by the use of torsion point images and the prime is changed to one of the form $p = 2^r \ell_1 \cdots \ell_n - 1$, for some large enough exponent r. In particular, the torsion points used in the protocol have order 2^r. The inclusion of these torsion points might seem suspicious especially since the recent polynomial-time attacks on SIDH explicitly use the torsion points. However, in this scheme, we are only given one torsion point image rather than the image of a full basis. Also, the degree of the isogeny used in CSIDH/SiGamal/SimS is not publicly revealed, unlike its SIDH counterpart. This makes running the attack in this setting rather difficult.

Cryptographic group actions. The idea behind this key exchange can be viewed in terms of abstract group actions. After all, this protocol specifically uses the CM action so one could replace this with any other action. In particular, if one replaces this group action with the group action that exponentiates a group element then you recover traditional Diffie-Hellman. The concept of cryptographic group actions will be explored further in Chapter VII.

CSIDH on the surface. In the original description of CSIDH, the $\mathbb{F}_p$-rational endomorphism ring is fixed to be $\mathbb{Z}[\pi]$. However, when $p = 3 \mod 4$, one can also have supersingular elliptic curves whose $\mathbb{F}_p$-rational endomorphism ring is $\mathbb{Z}[(1+\pi)/2]$ and do the CSIDH style protocol but using these supersingular curves. This was explored by Castryck and Decru in their protocol CSURF [CD20] and requires to select a prime which is of the form $p = 8\ell_1 \cdots \ell_n - 1$.

Signatures Schemes. There has been work constructing signature schemes based on the group action used in CSIDH [DFG19a, DPV19, BKV19]. The underlying idea here is to construct an identification protocol similar in design to the graph isomorphism identification protocol. Here one constructs a commutative diagram of isogenies and reveals certain ideals within the diagram depending on some bit challenge. Doing this solely with the ideals mentioned in Section IV.5.1 is not straightforward since there would be some leakage of the secret ideal as part of the interaction. The solution adopted by [DFG19a, DPV19] uses rejection sampling in order to fix these leakages, but this is fairly expensive. The solution adopted in [BKV19] completely avoids the need for rejection sampling by doing an expensive precomputation of the underlying class group. One can then sample an element of the class group uniformly at random. While this makes for a faster signature algorithm, this precomputation was only done for the CSIDH-512 parameter set and it does not scale to large parameters. This is because the class group precomputation requires an algorithm which takes subexponential complexity which is expensive for these larger parameters. As part of recent work [FFK$^+$23], an alternative approach has been explored that makes it possible to scale this class group precomputation slightly but results in very slow performance benchmarks and hence would not currently be considered for practical purposes.

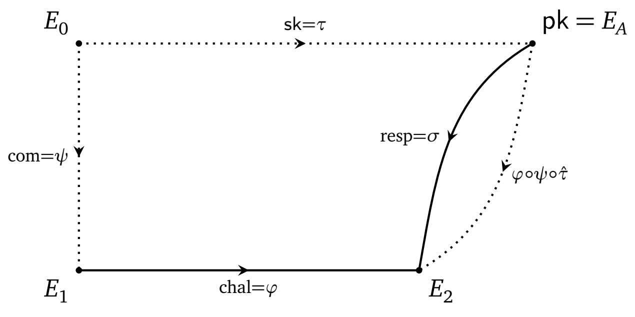


Figure 7: The SQISign identification protocol. The dotted lines indicate that the isogeny between the curves is kept secret and otherwise the isogeny is publicly known.

IV.5.3 Cryptanalysis

The isogenies between E and $E_{\mathfrak{a}}, E_{\mathfrak{b}}$ are $\mathbb{F}_p$-rational. So the fundamental hard problem that underpins the security of CSIDH is a restricted case of Problem 1 where one is tasked to recover an $\mathbb{F}_p$-rational isogeny. As already mentioned in IV.2.1, the best algorithm to find such an isogeny classically uses $\tilde{O}(p^{1/4})$ time and quantumly takes subexponential time. The quantum subexponential algorithm is related to the Kuperberg's hidden shift algorithm [Kup05] which can be adapted to these isogenies through the class group action [CJS14, BJS14]. Recent analysis of Kuperberg's algorithm applied to CSIDH [BS20, Pei20] suggests the parameters first proposed for CSIDH [CLM+18] fall well short of their intended quantum security target and thus one would need to increase the size of the parameters in order to counter this attack.

Outside of the techniques mentioned above, there are no other known methods to solve the underlying CSIDH problem.

IV.6 SQISign

SQISign (short quaternion isogeny signatures) is a high-soundness one-round identification protocol which is turned into a signature scheme using the Fiat Shamir (or Unruh) transform. It was proposed by De Feo, Kohel, Leroux, Petit and Wesolowski [DFKL+20] and is inspired by the GPS signature scheme by Galbraith, Petit and Silva [GPS17] whereby one adopts the KLPT algorithm to compute new isogenies between two supersingualar curves when given their endomorphism rings. The resulting isogeny is then used to generate the signature. We now give a high level description of the signature scheme as originally formulated in [DFKL+20] as well as give some recent developments in making the scheme faster. Figure 7 summarises the identification protocol that underpins SQISign.

IV.6.1 Identification Protocol

Choose a prime p such that $p^2 - 1 = \ell^f TR$ where ℓ is a small prime (typically $\ell = 2$), f is as large as possible, $T \approx p^{3/2}$ is not only coprime to ℓ but is also as smooth as possible and R is the remaining cofactor that could include some rough factors. Decompose $T = D_c T'$ where $D_c \approx \sqrt{p}$ and $T' \approx p$. Let $E_0/\mathbb{F}_p$ be a supersingular elliptic curve that has a known special extremal endomorphism ring $\mathrm{End}(E) \cong \mathcal{O}_0$. The term special extremal was introduced by the authors of the original KLPT algorithm [KLPT14] and refers to the fact that within the quaternion algebra, $\mathbb{Q}[i, j]$, for which $\mathrm{End}(E_0)$ lies in, this endomorphism ring has a suborder that admits an orthogonal decomposition of the form $R + jR$ where $R \subseteq \mathbb{Q}[i]$ is a quadratic order. Finally, let $N \approx p^{1/4}$ be a random prime number that is inert in R.

The key generation algorithm consists of computing a random isogeny of degree N. As we have seen already, computing this isogeny directly from Vélu is not feasible but since E_0 is special extremal then we can use the constructive Deuring correspondence to compute the isogeny. Call this isogeny $\tau : E_0 \to E_A$ where E_0 is the codomain supersingular curve. Then the public key and private key pair is $(\mathsf{pk}, \mathsf{sk}) = (E_A, \tau)$.

Now we look at the identification protocol itself. We recall that this first consists of an interaction between a prover and a verifier which has three stages: a commitment, followed by a challenge and finished off with a response.

Commitment: The prover generates a random isogeny $\psi : E_0 \to E_1$ of degree T' again using the constructive Deuring correspondence. The prover keeps the description of ψ secret and sends the curve E_1 to the verifier.

Challenge: The verifier receives the committing supersingular curve E_1 and computes a cyclic isogeny $\varphi : E_1 \to E_2$ of degree D_c. The verifier sends the complete description of φ to the prover.

Response: The prover receives the isogeny φ and from the isogeny composition $\varphi \circ \psi \circ \hat{\tau} : E_A \to E_2$ computes a new isogeny $\sigma : E_A \to E_2$ using the KLPT algorithm whose degree is some large power of the small prime ℓ such that $\hat{\varphi} \circ \sigma$ is cyclic. This power will be larger than the accessible ℓ^f so rather giving the verifier the entire description of σ, it returns a chain of ℓ^f-isogenies whose composition is σ.

The verification of this interaction consists of checking that the resulting isogeny σ is actually an isogeny from E_A to E_2 and that the composition $\hat{\varphi} \circ \sigma$ is cyclic. If it is then it accepts and otherwise it rejects.

As already mentioned previously, one can turn this into a signature scheme by applying the Fiat Shamir transform to the above identification protocol.

Modified KLPT algorithm. We note that the KLPT algorithm from [KLPT14] is slightly different than the adopted KLPT algorithm used here. In the process of obtaining the isogeny σ, the original algorithm ends up revealing some path from E_A to E_0 and thus would reveal the secret isogeny τ. Hence, they use a modified and generalised version of the algorithm that uses Eichler orders in order to compute a secure isogeny σ. The details of this is beyond the scope of this text and we refer to [DFKL+20, Section 5] for more details on this.

Recent parameter setup. As part of more recent developments related to this generalised KLPT algorithm, it is possible to reduce the size of the smooth integer T from $p^{3/2}$ to $p^{5/4+\epsilon}$. This is work due to De Feo, Leroux, Longa and Wesolowski [DFLLW23]. The necessity ϵ here refers to the fact that, if this cofactor is too close to $p^{5/4}$, then the underlying heuristics within the generalised KLPT algorithm might fail and one cannot guarantee a successful signature in SQISign [DFLLW23, Section 3.2]. Thus, in practice we need ϵ to be not too small (e.g., $0.02 < \epsilon < 0.1$). Again we refer the reader to [DFLLW23] for the specific details of this but note that one can obtain faster signatures with this approach especially when one incorporates recent fast algorithms for arithmetic over large prime fields [Lon22].

Using higher-dimensional isogenies. More recently, a new signature scheme has been proposed which is based on the SQISign framework and exploits the attacks on SIDH that use higher-dimensional isogenies as a tool to verify the signature [DLRW23]. This scheme is referred to as SQISignHD. In particular, they propose to use isogenies in dimension 4 and 8 as part of the verification procedure in SQISignHD. Additionally, they are able to exploit SIDH-style primes as part of their parameter setup and the signature algorithm consists of perform 2 and 3-isogenies. This is an improvement to the original SQISign design and suggests the signing can be made a lot faster. Having said this, there is no implementation of this scheme and it remains unclear whether the higher-dimensional isogenies required for verification can be implemented efficiently enough to consider SQISignHD for practical applications.

IV.6.2 Security Analysis

Fundamental to the security of SQISign is the endomorphism ring problem. If one can efficiently compute the endomorphism ring of a supersingular elliptic curve, then with the knowledge of $\mathrm{End}(E_A)$ and $\mathrm{End}(E_2)$ one compute the isogeny σ. Thus could one can easily forge a signature if the endomorphism ring problem is no longer intractable. Moreover, all isogenies in the protocol can be efficiently computed given the knowledge of their endomorphism rings.

More concretely, with respect to the identification protocol, the soundness security can be reduced to the one endomorphism problem. If the one endomorphism problem is hard then the identification protocol satisfies the special soundness property. The zero-knowledge property is technical and requires a new ad hoc assumption. Essentially, the assumption says that no polynomial-time adversary should be able to distinguish an isogeny σ resulting from the generalised KLPT algorithm from a random isogeny of the same degree, with non-negligible probablity. We refer to [DFKL$^+$20, DFLLW23] for the precise details of the assumption and some brief justifications as to why it is at least conjectured to be as hard as other isogeny problems such as the endomorphism ring problem.

Part 2

Contributions

Chapter V

Commitment Schemes from Supersingular Isogeny Graphs

In this chapter, we present work that was published in a special issue of the Journal of mathematical cryptography [Ste21].

V.1 Introduction

Commitment schemes [Blu83] have played a central role in the age of modern public-key cryptography. It allows a party to securely commit to particular value in such a way that other parties can be assured that it hasn't been tampered with. They have many useful applications: in secure electronic voting [CFSY96, DEG17], signature schemes [Lam79] and zero knowledge proofs [Dam98], to name a few.

One of the most important commitment schemes is one due to the Pedersen [Ped92] based on the hardness of the discrete logarithm problem in a finite cyclic group. As such, it is vulnerable to Shor's algorithm which renders it insecure if a sufficiently large quantum computer is available. Therefore, one might hope to design a commitment scheme which is secure against quantum adversaries. There has been some work on constructing lattice-based commitment schemes [XXW13, BDL+18]. They use well known lattice based assumptions such as Ring-LWE, Module-LWE and Module-SIS as a basis for their security. There has also been some work on constructing code-based commitment schemes [NTWZ19] and multivariate-based commitment schemes [PBB13].

At the time of writing, as far as we are aware, there are no published commitment schemes based on isogeny assumptions. Just as SIDH is an analogue of traditional Diffie-Hellman, one would hope that an analogue of Pedersen commitments exists in the isogeny setting. It is therefore surprising that this is not currently the case. Galbraith has declared this to be a “huge open problem" in isogeny-based cryptography [Gal20].

Contributions. In this chapter we present the first provably secure commitment schemes based on supersingular elliptic curve isogeny graphs. Underlying our protocols is the well-known idea of using a hash function to obtain a secure commitment scheme. In particular, we

```
$\mathsf{Exp}^{\text{hiding}}_{\mathscr{C},\mathscr{A}}(\lambda)$
1 : $\mathsf{pp} \leftarrow \mathsf{KeyGen}()$
2 : $(m_0, m_1) \leftarrow \mathscr{A}(\mathsf{pp})$
3 : $b \leftarrow\$ \{0,1\}$
4 : $r \leftarrow\$ \{0,1\}^{\lambda}$
5 : $c \leftarrow \mathsf{Commit}(\mathsf{pp}, m_b, r)$
6 : $b' \leftarrow \mathscr{A}(c)$
7 : return $b \stackrel{?}{=} b'$
```

```
$\mathsf{Exp}^{\text{binding}}_{\mathscr{C},\mathscr{A}}(\lambda)$
1 : $\mathsf{pp} \leftarrow \mathsf{KeyGen}()$
2 : $(m, m', r, r', c) \leftarrow \mathscr{A}(\mathsf{pp})$
3 : return $m \neq m'$
4 :         $\wedge\ \mathsf{Open}(\mathsf{pp}, m, r, c) \stackrel{?}{=} \mathsf{Open}(\mathsf{pp}, m', r', c)$
```

Figure 8: Hiding and binding experiments (resp.) for a commitment scheme.

use the CGL hash function described in Section IV.3 as a fundamental building block for our commitment scheme. The protocol requires a hard supersingular curve in order to make the construction secure. Hence a trusted setup as described in Section IV.2.2 is needed.

Traditionally, proving the security of the resulting commitment scheme is done with the help of the random oracle to show that it is information-theoretically hiding. However, in this work we obtain such a scheme without using the random oracle model: instead we use mathematical properties of isogenies and their associated isogeny graphs to obtain a commitment scheme which is both information-theoretically hiding and computationally binding.

Outline. In Section V.2 we begin with the necessary preliminaries needed for this work. This includes background on supersingular elliptic curve isogenies and we review the techniques used for computing such isogenies. We also give a formal definition of a commitment scheme and introduce the necessary security models. In Section V.3 we introduce the mixing constant for any regular graph and analyse its properties. In Sections V.4 and V.5 we present our commitment schemes based on supersingular isogeny graphs and use the result from Section V.3 to prove their security. In Section V.6 we estimate the performance of our commitment schemes, both from a perspective of efficiency and size of the commitment values. We also attempt to compare our schemes to that of a lattice counterpart. Finally, in Section V.7 we summarise the presented work and suggest avenues for future work.

V.2 Background on Commitment Schemes

Formally speaking, a commitment scheme consists of three algorithms: **KeyGen**(), **Commit**() and **Open**() - each of which has an implicit input 1^{λ} where λ is a security parameter. **KeyGen**() is a PPT algorithm that outputs the necessary public parameters needed for the protocol as well as the definition of the message space. **Commit**() is a PPT algorithm that, given the public parameters, a message m in the message space and a random $r \in \{0,1\}^{\lambda}$, outputs a value c which serves as the commitment to m and r. **Open**() is a deterministic polynomial-time algorithm that given the public parameters, the message m, the random r and the value c outputs a boolean value $b \in \{0,1\}$ according to whether or not c is a valid commitment to m and r.

Cryptographic applications of commitment schemes require the following two properties, known as hiding and binding. Informally, the hiding property ensures that the outputted commitment does not reveal anything about the message, while the binding property ensures that it should be hard to replicate the same commitment using a different message. We formally define these properties with aid of the games described in Figure 8. The hiding game is modelled like an indistinguishability game where the adversary is given the commitment of one of two messages and he is tasked to determine which message was used to derive the commitment. The binding game asks the adversary to find two distinct messages from your message space that gives the same commitment. Throughout these upcoming definitions, let $\mathscr{C}$ be a commitment scheme with a security parameter λ and $\mathscr{A}$ be an adversary.

Definition V.1. *The hiding advantage for the adversary $\mathscr{A}$, denoted $\mathsf{Adv}^{hiding}_{\mathscr{C},\mathscr{A}}(\lambda)$, is defined to be* $2\,|\Pr[\mathscr{A}$ *wins the hiding game*$] - 1/2|$. *More specifically, we have*

$$\mathsf{Adv}^{hiding}_{\mathscr{C},\mathscr{A}}(\lambda) = 2\left|\Pr\left[\mathsf{Exp}^{\text{hiding}}_{\mathscr{C},\mathscr{A}}(\lambda) = 1\right] - \frac{1}{2}\right|,$$

where $\mathsf{Exp}^{\text{hiding}}_{\mathscr{C},\mathscr{A}}(\lambda)$ *is the hiding experiment as defined in Figure 8. We say that $\mathscr{C}$ is information-theoretically (resp. computationally) hiding if for all adversaries (resp. PPT adversaries) $\mathscr{A}$ there is a negligible function,* negl, *such that the advantage of winning the hiding game is bounded above by* $\text{negl}(\lambda)$. *Furthermore we say $\mathscr{C}$ has perfect hiding if the hiding advantage is zero for any adversary.*

Lemma V.2. *Given a commitment scheme $\mathscr{C}$ and an adversary $\mathscr{A}$, we have*

$$\mathsf{Adv}^{hiding}_{\mathscr{C},\mathscr{A}}(\lambda) = \left|\Pr\left[\mathsf{Exp}^{\text{hiding},b=1}_{\mathscr{C},\mathscr{A}}(\lambda) = 1\right] - \Pr\left[\mathsf{Exp}^{\text{hiding},b=0}_{\mathscr{C},\mathscr{A}}(\lambda) = 1\right]\right|.$$

Definition V.3. *The binding advantage for the adversary $\mathscr{A}$, denoted $\mathsf{Adv}^{binding}_{\mathscr{C},\mathscr{A}}(\lambda)$, is defined to be* $\Pr[\mathscr{A}$ *wins the binding game*$]$. *More specifically, we have*

$$\mathsf{Adv}^{binding}_{\mathscr{C},\mathscr{A}}(\lambda) = \Pr\left[\mathsf{Exp}^{\text{binding}}_{\mathscr{C},\mathscr{A}}(\lambda) = 1\right],$$

where $\mathsf{Exp}^{\text{binding}}_{\mathscr{C},\mathscr{A}}(\lambda)$ *is the binding experiment as defined in Figure 8. We say that $\mathscr{C}$ is information-theoretically (resp. computationally) binding if for all adversaries (resp. PPT adversaries) $\mathscr{A}$ there is a negligible function,* negl, *such that the advantage of winning the binding game is bounded above by* $\text{negl}(\lambda)$. *Furthermore we say $\mathscr{C}$ has perfect binding if the binding advantage is zero for any adversary.*

V.3 Walking on Regular Graphs

Let G be a graph with vertex set $V(G)$ and let $(v_k)_{k\geq 0}$ denote a random walk in G. For a positive integer d (which throughout this work will always be at least 3), we say G is d-regular if for each vertex $v \in V(G)$ the number of edges incident to the vertex[4] v is d. We say the random walk (v_k) is non-backtracking if it does not traverse on the same edge twice in a row, i.e., for each $k \geq 1$ the edges $[v_{k-1}, v_k]$ and $[v_k, v_{k+1}]$ are different.

The adjacency matrix of the d-regular graph G, A, is the matrix whose (i, j)-th entry is the number of (directed) edges at the vertex i going to the vertex j. Note that the powers of this matrix describes the number of paths (that may include backtracking paths) between two

[4]If the graph G is directed then we specify that the number of outgoing edges from v is d.

vertices of a given length. The transition matrix of G, P, is the matrix whose (i,j)-th entry is $A(i,j)/d$. Finally, a stationary distribution on $V(G)$, π, is a probability distribution on the set of vertices of G such that $\pi = \pi P$ or equivalently $\pi(y) = \sum_{x\in V(G)} \pi(x)P(x,y)$. In effect, this distribution remains unchanged as you transition through the graph. If G is strongly connected, this stationary distribution is unique [LP17, Corollary 1.17].

Given a random walk (v_k) in G, we define the worst-case total-variation distance to stationarity at time t to be

$$d(t) := \frac{1}{2} \max_{v\in V(G)} \left\{ \sum_{x\in V(G)} \left| \Pr_v(v_t = x) - \pi(x) \right| \right\}$$

where $\Pr_v$ denotes the probability given $v_0 = v$ and π is the stationary distribution on G. We define $t_{\mathrm{MIX}}(\epsilon)$, the total-variation mixing time of (v_k) for $0 < \epsilon < 1$, as

$$t_{\mathrm{MIX}}(\epsilon) := \min\{t : d(t) < \epsilon\}.$$

Theorem V.4 (Rapid Mixing of Non-Backtracking Walks). *Let G be a random d-regular graph with N vertices and $d \geq 3$. Let (v_k) be a non-backtracking random walk in G. Then for any fixed $\epsilon > 0$, the worst case total-variation mixing time with high probability satisfies*

$$\begin{aligned} t_{\mathrm{MIX}}(1-\epsilon) &\geq \lceil \log_{d-1}(dN) \rceil - \lceil \log_{d-1}(1/\epsilon) \rceil, \\ t_{\mathrm{MIX}}(\epsilon) &\leq \lceil \log_{d-1}(dN) \rceil + 3\lceil \log_{d-1}(1/\epsilon) \rceil + 4. \end{aligned}$$

Proof. See [LS10, Theorem 2]. □

In other words, for a sufficiently small ϵ, this theorem says that the output of a non-backtracking random walk of length $O(\log_{d-1}(dN))$ on a random regular graph is indistinguishable from choosing a random vertex in the graph. It turns out that, compared to simple random walks that allow backtracking, the mixing time of non-backtracking walks is $\frac{d}{d-2}$ times smaller [LS10, Theorem 1].

In previous work including [JMV09], powers of the adjacency matrix, A^k, are used to get some mixing results on certain d-regular graph known as expander graphs. We are interested in the study of non-backtracking paths and for that we consider the following matrices: $A_1 = A$, $A_2 = A^2 - dI$ and $A_{r+1} = A_1 A_r - (d-1)A_{r-1}$ for $r \geq 2$. Then A_r is the matrix whose (i,j)-th entry is equal the number of non-backtracking walks from i to j of length r [Mur03, Section 6].

Lemma V.5. *Let G be a connected d-regular graph with $d \geq 3$. Then there exists some positive integer k_0 such that for all $k \geq k_0$, A_k has entries which are all non-zero.*

Proof. For any vertex i, the number of length r non-backtracking walks starting at i is precisely $d(d-1)^{r-1}$ and so we have

$$\sum_{k=1}^{\#V(G)} A_r(i,k) = d(d-1)^{r-1}.$$

Since $d \geq 3$, as $r \to \infty$ this sum tends to ∞ and hence there is some vertex j_0 such that $A_r(i,j_0) \to \infty$. For any vertex j, fix two paths (of length m_0, m_1) between $j_0 \to j$ (which can be done since the graph is connected). We ensure that the first step in these paths are different. Consider all paths $i \to j_0 \to j$ whereby we first go to j_0 and then traverse to j using one of our fixed paths making sure we avoid any backtracking. Then we have $A_r(i, j_0) \leq A_{r+m_0}(i, j) + A_{r+m_1}(i, j)$ and therefore $A_r(i, j) \to \infty$.

Hence for each i, j there is some $k(i, j)$ such that for all $k \geq k(i, j)$, we have $A_k(i, j)$ is strictly positive. Setting k_0 to be the maximum of $k(i, j)$ over all pairs of vertices (i, j) gives the result. □

Definition V.6. *We define k_G to be the minimal k_0 such that Lemma V.5 holds and call k_G the mixing constant for the graph G.*

The minimality of k_G means that there exists i_0, j_0 such that $A_{k_G-1}(i_0, j_0) = 0$, and for all i, j and $k \geq k_G$, $A_k(i, j) \neq 0$. Rephrasing this in the context of non-backtracking walks we obtain the following.

Corollary V.7. *For a connected d-regular graph G (with $d \geq 3$) let k_G be the corresponding mixing constant. Then for all $k \geq k_G$ and every pair of vertices (i, j), there exists a non-backtracking path between i and j of length k.*

We now provide a lower bound on the mixing constant k_G. The following is a generalisation of the calculation done in [ACNL+19, Section 6]. There are at most $d(d-1)^{k-1}$ possible outputs to a non-backtracking walk of length k. For some large enough k this number of walks exceeds the number of vertices in G: $d(d-1)^{k-1} \geq N$. Rearranging this gives us a lower bound for the mixing constant:

Lemma V.8. *The mixing constant k_G of a connected d-regular graph is bounded below by*

$$k_G \geq \log_{d-1}(N) - \log_{d-1}(d) + 1.$$

Theorem V.4 hints at an upper bound for k_G. Namely for a suitably small $\epsilon > 0$ we expect that $k_G \leq t_{\mathrm{MIX}}(\epsilon)$. In particular, if ϵ is negligibly small then the mixing constant may be at most $t_{\mathrm{MIX}}(\epsilon)$. For instance $\epsilon = 1/dN$, then by Theorem V.4 we get that $t_{\mathrm{MIX}}(1/dN) \leq 4\lceil \log_{d-1}(dN) \rceil + 4$. To summarise we make the following conjecture.

Conjecture V.9. *The mixing constant k_G of a connected d-regular graph G has the following upper bound:*

$$k_G \leq 4\lceil \log_{d-1}(dN) \rceil + 4.$$

This upper bound can be thought of as a worst case bound among all regular graphs. Some regular graphs have faster mixing rates, such as expander graphs or Ramanujan graphs, so one would hope expect that the mixing constant would be smaller. Later we conjecture better upper bounds for this mixing constant in the context of supersingular isogeny graphs as well as providing some experimental data to support the conjecture.

V.4 A Commitment Scheme from Isogeny Assumptions

The idea of using Ramanujan graphs, that have optimal mixing properties [Alo86], in cryptography was first proposed by [CLG09]. More precisely they proposed to construct hash functions by going on random walks on certain Ramanujan graphs where path-finding is hard. This includes supersingular isogeny graphs which were proved by Pizer [Piz90] to be Ramanujan.

In this section we use supersingular elliptic curve isogeny graphs to construct a commitment scheme and use the graph theoretic results from Section V.3 to prove its security. The idea behind our commitment scheme is, given a message m and a random r that someone wants to commit to, compute the isogeny-based hash of m concatenated by r. The output of this concatenation will be used as the commitment of the message m. Initially we present it in the supersingular 2-isogeny setting graph and later generalise it to the supersingular ℓ-isogeny with ℓ an odd prime.

V.4.1 Our Protocol

Let λ be a security parameter. The key generation of the commitment scheme is as follows. Choose a prime number p of 2λ bits and a positive integer k to be chosen later. This choice of p will ensure that our protocol will attain λ bits of classical security and $\lambda/2$ bits of quantum security against the Delfs-Galbraith and Biassse, Jao and Sankar (resp.) algorithms as mentioned in Section IV.2.1. In addition, with aid of a trusted setup, choose a hard supersingular curve $E/\mathbb{F}_{p^2}$. Recall from Section IV.2.2 that this means that no one should know nor be able to compute its endomorphism ring $\mathrm{End}(E)$. In addition, choose two random edges incident to $j(E)$ in the supersingular 2-isogeny graph.

To commit to a message $m \in \{0,1\}^k$ first compute the curve $E_m := \Phi_2(E, m)$ (making sure the first step in the graph is one of the two edges chosen above). Then choose uniformly at random a binary string $r \in_R \{0,1\}^k$ and compute the curve $E' := \Phi_2(E_m, r)$. When you go from E_m to E', make sure to avoid any backtracking in the isogeny graph. Then return $c := j(E')$ as the commitment of the message m.

Given the message m, the random r and the commitment c, to open the commitment first compute the curve $\Phi_2(\Phi_2(E, m), r)$. Then return the boolean value $c == j(\Phi_2(\Phi_2(E, m), r))$.

Remark V.10. *The necessity of the endomorphism ring of E remaining unknown is due to an attack by [EHL+18]. They are able to break the second preimage resistance of the isogeny hash function when the endomorphism ring of E is known. This will be important in the context of binding of our protocol.*

V.4.2 Hiding

The graph theoretic results presented in Section V.3 along with the following well known result on random walks on isogeny graphs will be used here to show that the commitment scheme presented in the previous subsection is information-theoretically hiding.

Theorem V.11. *Given a prime number p, let j_0 be a supersingular j-invariant in characteristic p, N_p be the number of supersingular j-invariants in characteristic p and $n = \prod_i \ell_i^{e_i}$ be an integer where ℓ_i are small primes. Let $\hat{j}$ be the j-invariant reached by a random walk of degree n starting at j_0. Then for every j-invariant $\tilde{j}$ we have*

$$\left|\Pr\left[\hat{j} = \tilde{j}\right] - \frac{1}{N_p}\right| \leq \prod_i \left(\frac{2\sqrt{\ell_i}}{\ell_i + 1}\right)^{e_i}.$$

Proof. See [GPS17, Theorem 1]. □

Theorem V.12. *Let $k_{2,p}$ be the mixing constant for the supersingular 2-isogeny graph in characteristic p. Then for any $k \geq k_{2,p}$, the commitment scheme described in Section V.4.1 is information-theoretically hiding.*

Proof. Fix two message strings m_0, m_1, a randomly chosen bit $b \in_R \{0,1\}$ and a resulting commitment $E' = \Phi(E_{m_b}, r)$. The goal for an adversary is to determine which message was used to get the commitment. Since the supersingular 2-isogeny graph is 3-regular, $k_{2,p}$ is well-defined. For any $k \geq k_{2,p}$, by Corollary V.7, there is guaranteed to be a path of length k from E_{m_0} to E' and E_{m_1} to E'. Set $\alpha := \frac{3}{2\sqrt{2}} > 1$. Using Theorem V.11 we have

$$\Pr\left[c = E' \mid \text{message is } m_0\right] - \Pr\left[c = E' \mid \text{message is } m_1\right] \leq 2\alpha^{-k}.$$

Similarly this difference is bounded below by $-2\alpha^{-k}$. Therefore the advantage of winning the hiding game is at most $2\alpha^{-k} \leq 2\alpha^{-k_{2,p}} \leq 2\alpha^{-2\lambda+\log_2(36)}$ (last inequality is a consequence of Lemma V.8 and $N_p \geq p/12 - 1$), which proves the theorem. □

By the conjectural upper bound on the mixing constant, Conjecture V.9, we can choose $k = 4\lceil \log_2(p) \rceil - 4$. With this choice of k we achieve information-theoretic hiding for our commitment scheme. As mentioned earlier, it could be possible to improve on this choice of k when specific graphs are used. Since supersingular isogeny graphs are Ramanujan graphs, one hopes that the mixing constant for these graphs is smaller. In particular we conjecture the following upper bound which we believe to be sharp for supersingular 2-isogeny graphs.

Conjecture V.13. *With $k_{2,p}$ be as defined previously, we have the following upper bound*

$$k_{2,p} \leq \log_2(p) + \log_2(\log_2(p)) + O(1).$$

In particular the constant in the big-Oh notation is at most 1.

Experimental results on this conjecture show that for every prime $p \leq 65600$ and some primes between $123000 \leq p \leq 131100$ and $234000 \leq p \leq 2^{18}$, the associated mixing constant for the supersingular 2-isogeny graph is no more than $\log_2(p)+\log_2(\log_2(p))+\frac{3}{10}$. These mixing constants were calculated by first computing the adjacency matrix, A, for the graph and sequentially computing A_k, as defined in Section V.3, until you find a value $\hat{k}$ such that the each entry of $A_{\hat{k}}$ is non-zero. We verify that $k_{2,p} = \hat{k}$ by computing $A_{\hat{k}+1}, A_{\hat{k}+2}, \cdots, A_{\hat{k}+i}$ for some small i and see if the entries in these matrices are non-zero. Since the entries of these matrices grow as we increase k, then as long as these matrices have non-zero entries, we can conclude that $k_{2,p} = \hat{k}$. Figure 9 tabulates the mixing constant in the supersingular 2-isogeny graph in characteristic p for all $p \leq 65600$. These experiments were conducted in Magma on a machine configured with two Xeon E5-2667v3 3.20 GHz, 378GB of RAM. The computation used 32 parallel threads and took one whole day to run in its entirety.

If this conjecture is true then we can choose $k = \lceil \log_2(p) + \log_2(\log_2(p)) + 1 \rceil$ and it would significantly speed up the performance of the protocol.

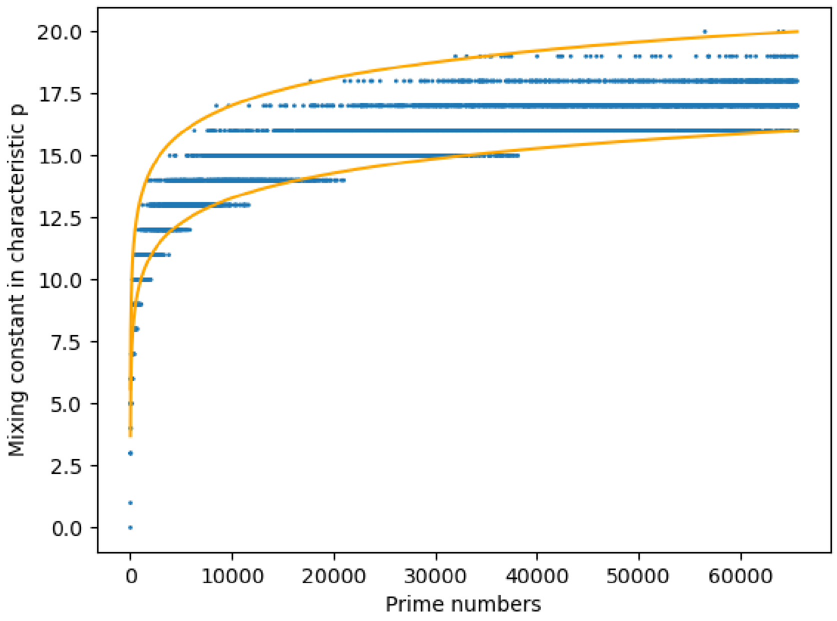


Figure 9: Mixing constant $k_{2,p}$ in prime characteristic p for all $p \leq 65600$. The lower bound curve is $\log_2(x)$ and the upper bound curve is $\log_2(x) + \log_2(\log_2(x))$.

V.4.3 Binding

Recall the "one supersingular endomorphism problem" as defined in Problem 4. In that problem one is tasked to compute a non-trivial and cyclic endomorphism of a supersingular curve. In its full generality, the degree of this endomorphism which is tasked to compute is in general smooth. We will prove that the binding of our protocol is secure under the hardness of this problem when specifically tasked to find an endomorphism whose degree is some power of 2.

Theorem V.14. *The commitment scheme as described in Section V.4.1 is computationally binding under the "one supersingular endomorphism problem" on the curve E – finding an endomorphism whose degree is a power of $\ell = 2$.*

Proof. Suppose that $\mathscr{A}$ is a PPT adversary which successfully solves the binding game for this commitment scheme. We shall construct an PPT adversary $\mathscr{A}'$ using $\mathscr{A}$ as a black box that solves the Supersingular Smooth Endomorphism Problem on the curve E.

Upon receiving the curve E, $\mathscr{A}'$ queries $\mathscr{A}$ and successfully outputs m, m', r, r' such that $m \neq m'$ and $E' = \Phi_2(E_m, r) = \Phi_2(E_{m'}, r')$. Let $\phi_m : E \to E_m$, $\phi_{m'} : E \to E_{m'}$, $\phi_r : E_m \to E'$, $\phi_{r'} : E_{m'} \to E'$ be the associated isogenies each of which has degree 2^k.

Then the composition of $\phi_m \circ \phi_r \circ \hat{\phi}_{r'} \circ \hat{\phi}_{m'}$ is an endomorphism of E whose degree is the prime power 2^{4k}. First we need to verify that this composition is non-trivial. Suppose for a contradiction that $\phi_m \circ \phi_r \circ \hat{\phi}_{r'} \circ \hat{\phi}_{m'} = [2^{2k}]$. Since the compositions $\phi_m \circ \phi_r$ and $\phi_{m'} \circ \phi_{r'}$ are isogenies from E to E' of same degree, then $\hat{\phi}_{r'} \circ \hat{\phi}_{m'}$ is the dual of $\phi_m \circ \phi_r$. Since $\widehat{\phi_{m'} \circ \phi_{r'}} =$

$\hat{\phi_{r'}} \circ \hat{\phi_{m'}}$, we get $\phi_m \circ \phi_r = \phi_{m'} \circ \phi_{r'}$. As a result $m = m'$ and $r = r'$ – which gives the contradiction.

By removing any potential backtracking to the composition $\phi_m \circ \phi_r \circ \hat{\phi_{r'}} \circ \hat{\phi_{m'}}$ that might occur as we approach E', we get a cyclic endomorphism ψ. The adversary $\mathscr{A}'$ outputs this endomorphisms and solves the Supersingular Smooth Endomorphism Problem on E in PPT. Therefore the advantage of winning the binding game is at most the advantage of solving the above problem. □

V.4.4 Generalisation

In this section we generalise the above idea and construct a commitment scheme which works in the supersingular ℓ-isogeny graph for a small odd prime ℓ. Once again, key generation of the protocol is the same as described in Section V.4.1.

To commit to a message $m \in \{0, 1, \cdots, \ell - 1\}^k$ first compute the curve $E_m := \Phi_\ell(E, m)$. Then choose uniformly at random a binary string $r \in_R \{0, 1, \cdots, \ell - 1\}^k$ and compute the curve $E' := \Phi_\ell(E_m, r)$. Once again, when you go from E_m to E', making sure to avoid any backtracking in the isogeny graph. Then return $c := j(E')$ as the commitment of the message m.

Given the message m and the random r, to open the commitment scheme first compute the curve $\Phi_\ell(\Phi_\ell(E, m), r)$. Then return the boolean value $c == j(\Phi_\ell(\Phi_\ell(E, m), r))$.

Much like in the setting of the 2-isogeny graph, we have the following theorems proving the security of this commitment scheme.

Theorem V.15. *Let $k_{\ell,p}$ be the mixing constant for the supersingular ℓ-isogeny graph in characteristic p. Then for any $k \geq k_{\ell,p}$, the commitment scheme described above is information-theoretically hiding.*

Proof. The proof is analogous to the proof of Theorem V.12. □

Much like in Section V.4.2, choosing $k = 4\lceil \log_\ell(p) \rceil + 8$ would be sufficient to get information-theoretic hiding.

Theorem V.16. *The above commitment scheme is computationally binding under the "one supersingular endomorphism problem" on the curve E – finding an endomorphism whose degree is a power of the odd prime ℓ.*

Proof. The proof is analogous to the proof of Theorem V.14. □

Along with this we make the following conjecture on a sharp upper bound of the mixing constant in these graphs.

Conjecture V.17. *With $k_{\ell,p}$ as above, we have*

$$k_{\ell,p} \leq \log_\ell(p) + \log_\ell(\log_\ell(p)) + O(1).$$

In particular the constant in the big-Oh notation is at most 1.

V.5 Commitments from Kernel Subgroups

In this section we describe a variant of the protocol from the previous section which uses the SIDH framework. Instead of using SIDH friendly primes we use primes of the form $2^n f - 1$ and achieve the same security requirements that were achieved in the previous section. One advantage of doing this is to exploit SIDH strategies [DFJP14, Section 4.2.2] to speed up isogeny computations. (Similar ideas in the context of the hash function construction can be found here [DPB17]). It is worth mentioning that while we base this on the SIDH framework, there are no torsion point images that are revealed so there is no impact on its security from the genus two setting

Let $p = 2^n f - 1$ be a prime with 2λ bits and f is a small integer. In the same manner as described in the previous section, choose a supersingular elliptic curve $E/\mathbb{F}_{p^2}$ whose endomorphism ring is unknown but this time we make sure that $\#E(\mathbb{F}_{p^2}) = (2^n f)^2$. This is done intentionally so that the 2^n-torsion subgroup of E entirely consists of points whose coordinates are in $\mathbb{F}_{p^2}$. Let $P_0, P_1 \in E[2^n]$ be points on E that form a basis for this 2^n-torsion subgroup of E.

Much like in the Section V.4, we will go on walks in the supersingular 2-isogeny graph but instead of choosing at each step which edge to traverse, we compute the kernel subgroup and corresponding the isogeny whose kernel is this subgroup. However the longest isogeny that can be computed as a sequence of 2-isogenies using this approach has degree 2^n. So in order to attain the same security as a obtained in Section V.4.2, namely computing an isogeny of degree 2^k with $k = 4\lceil \log_2(p) \rceil \approx 4n$, we must do this isogeny computation 4 times each for the message used and the randomly generated element.

Recall that given $m_0 \in \mathbb{Z}/2^n\mathbb{Z}$ the subgroup of $E[2^n]$ defined by $\langle P + m_0 Q \rangle$ induces an isogeny $\phi_{m_0} : E \longrightarrow E_{m_0}$ whose kernel is this subgroup. If we wish to extend this walk by going on another walk of degree 2^n, then we must find points P', Q' on E_{m_0} that form a basis for the respective 2^n-torsion subgroup. Also we need a procedure of computing these points in a deterministic manner. Ensuring that if replicated by another party we get the same points. We already know that $\phi_{m_0}(Q)$ has order 2^n, so set $Q' := \phi_{m_0}(Q)$.

To deterministically compute a point P' we use techniques from [CJL+17]. We briefly summarise this method. Use the Elligator 2 method for deterministically computing points R in $E(\mathbb{F}_{p^2})$ [BHKL13], then check that $R \in E \setminus [2]E$, where $[2]E$ is the set of all 2-divisible points on E. If so then the point fR is a point of order 2^n. A check has to be made to see if this point is independent from Q'. If not then it cannot be used as a second basis element and we repeat the whole process until you compute a point P' which can be used as the second basis element. For more details on this see [CJL+17, Section 3.2].

Remark V.18. *The choice of $Q' = \phi_{m_0}(Q)$ was done purposefully. It ensures that the isogeny induced by a kernel of the form $\langle P' + m_1 Q' \rangle$ will not result in backtracking through part of the first isogeny. This is because the kernel of the dual isogeny, $\ker(\widehat{\phi_{m_0}})$, is generated by the point Q' [NR19, Proposition 3].*

V.5.1 Protocol Description and Security

The key generation is as described above.

To commit to a message $m \in \mathbb{Z}/2^{4n}\mathbb{Z}$ do as follows. Compute $m_0 := m \mod 2^n$, $m_1 := (m - m_0)/2^n \mod 2^n$, $m_2 := (m - m_1 2^n - m_0)/2^{2n} \mod 2^n$ and $m_3 := (m - m_2 2^{2n} - m_1 2^n - m_0)/2^{3n} \mod 2^n$. Notice that $m_0, m_1, m_2, m_3 \in \mathbb{Z}/2^n\mathbb{Z}$. Compute the subgroup $M_0 := \langle P + m_0 Q \rangle$ and hence the corresponding isogeny $\phi_{m_0} : E \to E_{m_0}$ whose kernel is M_0. Compute the point $Q' := \phi_{m_0}(Q)$ and a point P' as described above. Now compute the subgroup $M_1 := \langle P' + m_1 Q' \rangle$ and hence the corresponding isogeny $\phi_{m_0} : E_{m_0} \to E_{m_1}$ whose kernel is M_1. Again compute the points P'', Q''. Repeat this for the integers m_2, m_3 to get isogenies $\phi_{m_i} : E_{m_{i-1}} \to E_{m_i}$ whose kernel is M_i $(i = 2, 3)$. Henceforth the curve $E_m := E_{m_3}$ and the composition $\phi_m = \phi_{m_3} \circ \phi_{m_2} \circ \phi_{m_1} \circ \phi_{m_0}$ is the curve and isogeny obtained from the message m.

As remarked above, the choice of basis points for the 2^n-torsion subgroup is done so that we don't get any backtracking in the isogeny graph.

From here choose a random $r \in \mathbb{Z}/2^{4n}\mathbb{Z}$ and repeat the same procedure as done above. Once again you make sure that there is no backtracking through ϕ_m by making sure that you have an appropiate basis for the 2^n-torsion subgroup. The result is an isogeny $\phi_r : E_m \to E'$. Then return the curve $c := j(E')$ as the commitment of the message m.

Much like in Section V.4, given the message and the random $m, r \in \mathbb{Z}/2^{4n}\mathbb{Z}$ (resp.), to open the commitment you recompute the curve E' and return the boolean value $c == j(E')$. The deterministic nature of computing the new basis for the next 2^n-torsion subgroup means that, as long as the message m and the random r are as intended, then anyone can open the message and be assured that this is the correct message used.

Theorem V.19. *The commitment scheme described above is information theoretically hiding and computationally binding under the Supersingular Smooth Endomorphism Problem on the curve E for the prime $\ell = 2$.*

Proof. Application of Theorem V.12 & Theorem V.14. □

Remark V.20. *The idea of composing rational isogenies in a manner described above has recently been exploited in a new construction of an oblivious PRF [Bas23]. We note that the term irrational isogeny is used for the composition of rational isogenies, denoting the fact that the composed isogeny has a kernel that is no longer rational.*

V.6 Comparison

In this section we estimate the performance of these schemes, only in the setting when $\ell = 2$, and attempt to compare them to other post-quantum commitment schemes.

In the work by [DPB17], they attempted to compare the performance of the CGL hash function with a hash function that is analogous to the idea presented in Section V.5. If a prime of the form $p = 2^n f - 1$ is used and k is the length of the walk you want to compute, they estimated the complexity of the CGL hash function as $kn(5.7n + 110)\mathbf{m}$ and the complexity of the SIDH variant as $kn(13.5\log(n) + 42.4)\mathbf{m}$, where $\mathbf{m}$ is the cost of performing a field multiplication. These performance timings translate to our commitment scheme constructions by choosing $k = 4\lceil \log_2(p) \rceil - 4$ with one exception. In the SIDH variant of our commitment scheme a little more work is needed then that presented above since we need to generate the basis elements for the new torsion subgroup. This requires computing one isogeny image as well as the cost of doing the Elligator 2 method to determine the second basis element. Since this is done at most 3 times, it doesn't add much to the complexity mentioned above. Approximately it adds $O(n\mathbf{m})$ to the overall complexity which is primarily dominated by the isogeny image computations.

Therefore, the performance ratio of the scheme described in Section V.4 versus that of this Section V.5 is approximately $(5.7n + 110)/(13.5\log(n) + 46.4 + O(1/k))$. This implies an exponential speed up in the performance of the commitment scheme presented in this Section V.5 versus that described in Section V.4 (especially when the prime p is of cryptographic size).

As mentioned earlier, if the validity of Conjecture V.13 holds then the performance of these procotols will significantly speed up by up to a factor of 4.

Lets now look at the size of the commitment values in our schemes. In both variants, these values just consist of one j-invariant of a supersingular elliptic curve which is an element in $\mathbb{F}_{p^2}$. Equivalently, given a 2-dimensional representation of $\mathbb{F}_{p^2} = \mathbb{F}_p[i]$, we can express this j-invariant as two $\mathbb{F}_p$ elements. Hence, given a prime p of 2λ bits with λ a security parameter, the size of the commitment value is approximately 4λ bits or $\lambda/2$ bytes. It is worth mentioning that the size of the commitment value does not depend on the size of k. This point is consistent with most isogeny schemes, including the CGL hash function.

One can compare these commitment scheme to that of other post-quantum alternative. One clear advantage this has over other alternatives is that the size of the committed values. To target 128 bits of security, the size of the committed value in our scheme is approximately 64 B. In comparison to that of lattice based commitment schemes taken from [BDL+18, Table 2], to achieve the same level of security, the committed values is approximately 9 kB. This is much larger than that of our isogeny commitment schemes. There are a few notable drawbacks when comparing our schemes to its alternatives. First one is the performance of our schemes. Even faster variant described in Section V.5 is not as fast as its lattice counterpart. This point is again consistent with most isogeny schemes. Second drawback is that it is not a homomorphic commitment scheme. Having a homomorphic commitment scheme is desirable to have since there are some strong applications that rely on the homomorphic property such as electronic voting.

V.7 Conclusion

In this work we presented two commitment schemes based on isogeny assumptions. This is the first provably secure commitment scheme in the isogeny literature. The scheme follows the approach of [CLG09] whereby we go on walks in supersingular isogeny graphs. We proved that this commitment scheme is secure attaining information-theoretic hiding and computational binding. We obtained information-theoretic hiding based on the existence of a mixing constant, k_G, implying that any two vertices in the graph can be connected by a non-backtracking path of fixed length k for any $k \geq k_G$. We conjectured an upper bound on this constant for both the generic setting and the specific setting of supersingular isogeny graphs. We obtained computationally binding by reducing a binding instance to a well known isogeny problem which is believed to be hard even for quantum adversaries.

We also presented a variant of this commitment scheme which is constructed through a kernel subgroup to compute the isogenies instead of going through step by step and choosing which edge to continue. Its security follows directly from the security of the previous scheme. The main advantage that this variant has over the previous commitment scheme is that of efficiency.

There are a number of open problems that arise from this work.

- Proving the explicit upper bounds for the mixing constant in both the generic setting and the special setting of the supersingular isogeny graphs.
- See how sharp we can makes these upper bounds and see if we can get close to the bound presented in Conjecture V.13 and Conjecture V.17 in the specific setting of supersingular isogeny graphs.
- Constructing a homomorphic commitment scheme based on isogeny assumptions. This problem would be considered a major breakthrough in this area.

Chapter VI

Constructing Twin Smooth Integers with Applications to SQISign

In this chapter, we present an extension of the work that was done in collaboration with Giacomo Bruno, Maria Corte-Real Santos, Craig Costello, Jonathan Komada Eriksen, Michael Meyer and Michael Naehrig [BSC$^+$22] which has been accepted to Asiacrypt23. In particular, the sections that feature new and original work which is not presented in the corresponding article will be presented in Sections VI.6 and VI.7.

VI.1 Introduction

In recent years the tantalising problem of finding two large twin smooth integers has emerged in the context of instantiating efficient isogeny-based public key cryptosystems. Recall that this means we want to find consecutive integers, $(r, r+1)$, that are both smooth. As mentioned already in Chapter IV, this problem was initially motivated by key-exchange protocol B-SIDH (before the wave of polynomial time attacks surfaced) but has seen other more recent applications including the isogeny-based signature scheme is SQISign [DFKL$^+$20]; it boasts the smallest public keys and signatures of all post-quantum signature schemes (by far!), at the price of a signing algorithm that is orders of magnitude slower than its post-quantum counterparts. Finding secure parameters for SQISign is related to the twin smooth problem mentioned above, with a large contributing factor to the overall efficiency of the protocol being the smoothness bound, B, of the rational torsion used in isogeny computations. This bound corresponds to the degree of the largest prime-degree isogeny computed in the protocol, for which the VéluSqrt algorithm mentioned in Chapter IV is the fastest algorithm for computing such isogenies. Part of the reason for SQISign's performance drawback is that the problem of finding parameters with small B is difficult: the fastest implementation to date targets security comparable to NIST Level I [The16, Section 4.A] and has $B = 3923$ [DFLLW23]. Additionally, methods for finding efficient SQISign parameters have to date not been able to obtain suitable primes reaching NIST Level III and V security. In view of NIST's recent call for additional general purpose post-quantum signature schemes that are not based on structured lattices [The22], it is important to find methods of generating efficient isogeny-based signature parameters beyond those that have been proposed thus far at NIST Level I.

To date there have been many methods proposed to find such twin smooth integers. Among these known methods, they can be categorised into two groups: the first are *constructive* methods that seek to find all or almost all twin smooth integers for a given smoothness bound; and the second are *probabilistic* methods that will only guarantee finding twin smooth integers with some probability. An example of a probabilistic method is based on the extended Euclidean (XGCD) algorithm over the integers (which will be detailed in Section VI.2). This algorithm has been used to find SQISign parameters including the current state-of-the-art NIST Level I prime mentioned above.

The CHM algorithm. In this work we introduce new ways of finding twin smooth instances based on the Conrey-Holmstrom-McLaughlin (CHM) "Smooth neighbors" algorithm [CHM13]. This algorithm is a constructive algorithm which for a fixed smoothness bound B attempts to find almost all B-smooth twins. The CHM algorithm starts with the set of integers $S = \{1, 2, \ldots, B - 1\}$ representing the smooth twins $(1, 2), (2, 3), \ldots, (B - 1, B)$, and recursively grows this set by constructing new twin smooth integers from unordered pairs in $S \times S$ until a full pass over all such pairs finds no new twins, at which point the algorithm terminates. Although the CHM algorithm is not guaranteed to find the set of all B-smooth twins, for moderate values of B it converges with the set S containing *almost all* such twins. The crucial advantage is that, unlike the algorithm of Lehmer [Leh64] that exhaustively solves $2^{\pi(B)}$ Pell equations to guarantee the full set of B-smooth twins, the CHM algorithm terminates much more rapidly. For example, in 2011 Luca and Najman [LN11] used Lehmer's approach with $B = 100$ to compute the full set of 13,374 twin smooths in 15 days (on a quad-core 2.66 GHz processor) by solving $2^{\pi(B)} = 2^{25}$ Pell equations, the solutions of which can have as many as 10^{10^6} decimal digits. The largest pair of 100-smooth twins they found were the 58-bit integers

$$\begin{aligned}166055401586083680 &= 2^5 \cdot 3^3 \cdot 5 \cdot 11^3 \cdot 23 \cdot 43 \cdot 59 \cdot 67 \cdot 83 \cdot 89, \text{ and}\\ 166055401586083681 &= 7^2 \cdot 17^{10} \cdot 41^2.\end{aligned}$$

In 2012, Conrey, Holmstrom and McLaughlin ran *their* algorithm on a similar machine to find 13,333 (i.e. all but 41) of these twins in 20 minutes [CHM13]. Subsequently, they set $B = 200$ and found a list of 346,192 twin smooths in about 2 weeks, the largest of which were the 79-bit integers

$$\begin{aligned}589864439608716991201560 &= 2^3 \cdot 3^3 \cdot 5 \cdot 7^2 \cdot 11^2 \cdot 17 \cdot 31 \cdot 59^2 \cdot 83 \cdot 139^2\\ &\qquad \cdot 173 \cdot 181, \text{ and}\\ 589864439608716991201561 &= 13^2 \cdot 113^2 \cdot 127^2 \cdot 137^2 \cdot 151^2 \cdot 199^2.\end{aligned}$$

Exhausting the full set of 200-smooth twins would have required solving $2^{\pi(200)} = 2^{46}$ Pell equations, which is pushing the limit of what is currently computationally feasible. The largest run of Lehmer's algorithm reported in the literature used $B = 113$ [Cos20, Section 5.3], which required solving 2^{30} Pell equations and a significant parallelised computation that ran over several weeks.

The largest set of 113-smooth twins found during that computation were the 75-bit integers

$$\begin{aligned}19316158377073923834000 &= 2^4 \cdot 3^6 \cdot 5^3 \cdot 7 \cdot 23^2 \cdot 29 \cdot 47 \cdot 59 \cdot 61 \cdot 73 \cdot 97 \cdot 103,\\ 19316158377073923834001 &= 13^2 \cdot 31^2 \cdot 37^2 \cdot 43^4 \cdot 71^4.\end{aligned}$$

Remark VI.1. *The above examples illustrate some important phenomena that are worth pointing out before we move forward. Observe that, in the first and third examples, the largest prime not exceeding B is not found in the factors of the largest twins. The largest 89-smooth twins are the same as the largest 97-smooth twins, and the largest 103-smooth twins are the same as the largest 113-smooth twins. In other words, increasing B to include more primes necessarily increases the size of the set of B-smooth twins, but it does not mean we will find any new, larger twins. This trend highlights part of the difficulty we face in trying to find optimally smooth parameters of cryptographic size: increasing the smoothness bound B makes the size of the set of twins grow rapidly, but the growth of the largest twins we find is typically painstakingly slow. The set of* 100-*smooth twins has cardinality 13,374, with the largest pair being 58 bits; increasing B to 200 gives a set of cardinality (at least) 345,192, but the largest pair has only grown to be 79 bits. In fact, most of this jump in the bitlength of the largest twins occurs when increasing* $B = 97$ *(58 bits) to include two more primes with* $B = 103$ *(76 bits). Including the 19 additional primes up to 199 only increases the bitlength of largest twins with* $B = 199$ *by 3 (79 bits), and this is indicative of what we observe when B is increased even further.*

Contributions. In this chapter we revisit this CHM algorithm in such a way so that we can run it for much larger values of B in order to find larger sized twins. For example, the original CHM paper reported that the full algorithm with $B = 200$ terminated in approximately 2 weeks; our implementation did the same computation in around 943 seconds on a laptop. Increasing the smoothness bound to $B = 547$, our implementation converged with a set of 82,026,426 pairs of B-smooth twins, the largest of which are the 122-bit pair $(r, r+1)$ with

$$\begin{aligned}r &= 5^4 \cdot 7 \cdot 13^2 \cdot 17^2 \cdot 19 \cdot 29 \cdot 41 \cdot 109 \cdot 163 \cdot 173 \cdot 239 \cdot 241^2 \cdot 271 \cdot 283\\ &\quad \cdot 499 \cdot 509, \qquad \text{and}\\ r+1 &= 2^8 \cdot 3^2 \cdot 31^2 \cdot 43^2 \cdot 47^2 \cdot 83^2 \cdot 103^2 \cdot 311^2 \cdot 479^2 \cdot 523^2.\end{aligned}$$

Although it remains infeasible to increase B to the point where the twins found through CHM are large enough to be used out-of-the-box in isogeny-based schemes (i.e. close to 2^{256}), we are able to combine the larger twins found through CHM with techniques from the literature in order to find much smoother sets of SQISign parameters. In this case we are aided by the requirements for SQISign, which permit us to relax the size of the smooth factor that divides $p^2 - 1$. The current state-of-the-art instantiation [DFLLW23] uses primes p such that

$$\ell^f \cdot T \mid (p^2 - 1),$$

where ℓ is a small prime (typically $\ell = 2$), where f is as large as possible, and where $T \approx p^{5/4}$ is both coprime to ℓ and B-smooth. For example, the original SQISign implementation [DFKL+20] used a 256-bit prime p such that

$$p^2 - 1 = 2^{34} \cdot T_{1879} \cdot R,$$

where T_{1879} is an odd 334-bit integer[5] whose largest prime factor is $B = 1879$, and R is the *rough* factor; a 144-bit integer containing no prime factors less than or equal to B. As another example, De Feo, Leroux and Wesolowski [DFLLW23, Section 5] instead use a 254-bit prime p with

$$p^2 - 1 = 2^{66} \cdot T_{3923} \cdot R,$$

where T_{3923} is an odd 334-bit integer whose largest prime factor is $B = 3923$, and where all of R's prime factors again exceed B.

During the search mentioned above that found the record 547-smooth twins, over 82 million other pairs of smaller sized twins were found. One such pair was the 63-bit twins $(r-1, r)$ with $r = 8077251317941145600$. Taking $p = 2r^4 - 1$ gives a 253-bit prime p such that

$$p^2 - 1 = 2^{49} \cdot T_{479} \cdot R,$$

where T_{479} is an odd 328-bit integer that is 479-smooth. This represents a significant improvement in smoothness over the T values obtained in [DFKL$^+$20] and [DFLLW23]. Although the smoothness of T is not the only factor governing the efficiency of the scheme, our analysis in Section VI.5.3 suggests that the parameters found in this chapter compare favourably with those currently found in SQISign implementations.

Just as we transformed a pair of 63-bit twins into a 253-bit prime by taking $p = 2r^4 - 1$, we combine the use of twins found with CHM and primes of the form $p = 2r^n - 1$ with $n \geq 3$ to obtain several SQISign-friendly primes that target higher security levels. For example, with some 64-bit twins $(r, r+1)$ found through CHM, we give a 382-bit prime $p = 2r^6 - 1$ such that $p^2 - 1 = 2^{80} \cdot T_{10243} \cdot R$, where T is an odd 495-bit integer that is 10243-smooth; this prime would be suitable for SQISign signatures geared towards NIST Level III security. As another example, with some 85-bit twins $(r, r+1)$, we give a 508-bit prime $p = 2r^6 - 1$ such that $p^2 - 1 = 2^{86} \cdot T_{150151} \cdot R$, where T is a 639-bit integer that is 150151-smooth; this prime would be suitable for SQISign signatures targeting NIST Level V security.

Additionally, we detail a probabilistic method for generating these twin smooth integers that has not been presented in the literature. Inspired by the method of the XGCD algorithm over the integers mentioned earlier, this new method is based on this same algorithm but applied to the polynomial ring with rational coefficients. We show that this method is a generalisation of the known probabilistic methods that are based on polynomial evaluation and also highlight specific instances where this method is better compared to the prior methods.

We put this back into context of finding SQISign friendly parameters. For a particular choice of polynomials input to the XGCD algorithm, we can adopt a very similar strategy as we did previously. Namely take a twin found with CHM and feed it into the resulting polynomial obtained from the algorithm to hopefully give primes suitable for SQISign. The results from this experimentation give a competitive alternative to the primes mentioned above.

Organisation. In Section VI.2 we review prior methods for generating large twin smooth integers. In Section VI.3 we recall the CHM algorithm and give a generalisation of it that may be of independent interest. In Section VI.4 we detail our implementation of the CHM algorithm

[5]The initial SQISign requirements [DFKL$^+$20] had $T \approx p^{3/2}$, but T_{1879} corresponds to the new requirements.

and present a number of optimisations that allowed us to run it for much larger values of B. In Section VI.5 we discuss the combination of CHM with primes of the form $p = 2r^n - 1$ to give estimates on the probabilities of finding SQISign parameters at various security levels. In Section VI.5.3 we present our results, giving record-sized twin smooth instances as well as dozens of SQISign-friendly primes that target NIST's security levels I, III, and V. In Section VI.6 we describe this new method for finding twin smooth integers based on the XGCD algorithm over the polynomial ring $\mathbb{Q}[x]$. Finally in Section VI.7, we show how to combine the CHM with this new method to obtain SQISign-friendly primes. We describe the theory behind it as well as present the concrete primes that were found.

VI.2 Prior Methods of Searching for Large Twin Smooth Integers

We begin by reviewing prior methods of searching for twin smooth integers and follow the descriptions of the three algorithms reviewed in [CMN21, Section 2], as well as including the method introduced in [CMN21] itself.

Solving Pell equations. Fix B, let $\{2, 3, \ldots, q\}$ be the set of primes up to B with cardinality $\pi(B)$, and consider the B-smooth twins $(r, r+1)$. Let $x = 2r+1$, so that $x-1$ and $x+1$ are also B-smooth, and let D be the squarefree part of their product $(x-1)(x+1)$, i.e. $x^2 - 1 = Dy^2$ for some $y \in \mathbb{Z}$. It follows that Dy^2 is B-smooth, which means that

$$D = 2^{\alpha_2} \cdot 3^{\alpha_3} \cdot \cdots \cdot q^{\alpha_q}$$

with $\alpha_i \in \{0, 1\}$ for $i = 2, 3, \ldots, q$. There are a total of $2^{\pi(B)}$ possibilities for D. For each of these squarefree integers, Størmer [Stø97] reverses the above argument and proposes to solve the $2^{\pi(B)}$ Pell equations

$$X^2 - DY^2 = 1,$$

finding *all* of the solutions for which y is B-smooth, and in doing so finding the complete and finite set of B-smooth twins. Moreover, Lehmer [Leh64, Theorem 1] was able to quantify this result and proved the following.

Theorem VI.2. *Let $B > 2$, q be the largest prime that is at most B and $r \in \mathbb{Z}$ be a positive integer. Then the pair of consecutive integers $(r, r+1)$ are twin B-smooth integers if and only if there exists a B-smooth integer y, a square-free B-smooth integer D and an integer $1 \leq n \leq \max\{3, (q+1)/2\}$ such that the pair (x_n, y_n) with $x_n = 2m+1$ and $y_n = y$ is the n^{th} solution to the Pell equation*

$$X^2 - DY^2 = 1.$$

The largest pair of 2-smooth integers is $(1, 2)$, the largest pair of 3-smooth integers is $(8, 9)$, and the largest pair of 5-smooth integers is $(80, 81)$. Unfortunately, solving $2^{\pi(B)}$ Pell equations becomes infeasible before the size of the twins we find is large enough (i.e. exceeds 2^{200}) for our purposes. As we saw in Section VII.1, [Cos20] reports that with $B = 113$ the largest twins $(r, r+1)$ found upon solving all 2^{30} Pell equations have $r = 19316158377073923834000 \approx 2^{75}$. As a small remark, as part of recent work by Buzek et al. [BHL+22], they modified this approach to finding twin smooth integers that lie in an interval instead of collating all twin smooth integers.

The extended Euclidean algorithm. The most naïve way of searching for twin smooth integers is to compute B-smooth numbers r until either $r-1$ or $r+1$ also turns out to be B-smooth. A much better method [Cos20, DFKL$^+$20] is to instead choose two coprime B-smooth numbers α and β that are both of size roughly the square root of the target size of r and $r+1$. On input of α and β, Euclid's extended GCD algorithm outputs two integers (s,t) such that $\alpha s + \beta t = 1$ with $|s| < |\beta/2|$ and $|t| < |\alpha/2|$. We can then take $\{m, m+1\} = \{|\alpha s|, |\beta t|\}$, and the probability of m and $m+1$ being B-smooth is now the probability that $s \cdot t$ is B-smooth. The reason this performs much better than the naïve method above is that $s \cdot t$ with $s \approx t$ is much more likely to be B-smooth than a random integer of similar size.

Searching with $r = x^n - 1$. A number of works [Cos20, DFKL$^+$20, DFLLW23] have found performant parameters by searching for twins of the form $(r, r+1) = (x^n - 1, x^n)$, for relatively small $n \in \mathbb{Z}$. For example, suppose we are searching for b-bit twins $(r, r+1)$ and we take $n = 4$ so that $r = (x^2+1)(x-1)(x+1)$. Instead of searching for two b-bit numbers that are smooth, we are now searching for three smooth $(b/4)$-bit numbers (i.e. $x-1$, x, and $x+1$) and one smooth $(b/2)$-bit number, which increases the probability of success (see [CMN21]).

We give some notation to the result of summing these twins as it will be useful for later sections and set $p_n(x) := 2x^n - 1$

Searching with PTE solutions. The approach taken in [CMN21] can be viewed as an extension of the method above, where the important difference is that for $n > 2$ the polynomial $x^n - 1$ does not split completely in $\mathbb{Z}[x]$, and the presence of higher degree terms (like the irreducible quadratic $x^2 + 1$ above) significantly hampers the probability that values of $x^n - 1 \in \mathbb{Z}$ are smooth. Instead, the algorithm in [CMN21] takes $(r, r+1) = (f(x), g(x))$, where $f(x)$ and $g(x)$ are both of degree n and are comprised entirely of linear factors. This boosts the success probability again, but one of the difficulties facing this method is that polynomials $f(x)$ and $g(x)$ that differ by a constant and are completely split are difficult to construct for $n \geq 4$. Fortunately, instances of these polynomials existed in the literature prior to [CMN21], since they can be trivially constructed using solutions to the Prouhet-Tarry-Escott (PTE) problem. We briefly give an overview of the problem along with the connection to the polynomials that are sought after for constructing smooth twins.

Definition VI.3. *The Prouhet-Tarry Escott (PTE) problem of size n and degree k asks to find solutions $\{a_1, \cdots, a_n\}$ and $\{b_1, \cdots, b_n\}$ with $a_i \neq b_j$ for all i, j and that satisfies the following:*

$$\sum_{i=1}^{n} a_i^j = \sum_{i=1}^{n} b_i^j, \quad \textit{for } j = 1, \cdots, k.$$

When we have such a solution we call it a PTE solution of size n and degree k and sometimes abbreviate it to $[a_1, \cdots a_n] =_k [b_1, \cdots, b_n]$. In the case when $k = n-1$ then we call the solution an ideal PTE solution.

The following proposition shows how this problem is related to the polynomials which is desired [BLP03, Lemma 1].

Proposition VI.4. *Given the following distinct multisets of integers $\{a_1, \cdots, a_n\}$ and $\{b_1, \cdots, b_n\}$, we have $[a_1, \cdots a_n] =_k [b_1, \cdots, b_n]$ if and only if*

$$\deg\left(\prod_{i=1}^{n}(x-a_i) - \prod_{i=1}^{n}(x-b_i)\right) \leq n-(k+1).$$

In particular the solution is an ideal PTE solution if and only if the polynomials $\prod_{i=1}^{n}(x-a_i)$ and $\prod_{i=1}^{n}(x-b_i)$ differ by a constant.

VI.3 The CHM Algorithm

In this section, we recall the Conrey, Holmstrom, and McLaughlin (CHM) algorithm [CHM13], a remarkably simple algorithm that generates twin smooth integers, i.e. smooth values of the polynomial $X(X+1)$. We then present a generalisation of this algorithm, which generates smooth values of any monic quadratic polynomial. The algorithm generalises the CHM algorithm, as well as another algorithm in the literature by Conrey and Holmstrom [CH21], which generates smooth values of the polynomoial X^2+1. In the end, we are primarily interested in the CHM algorithm, but present the generalised algorithm here, as it may be of independent interest.

VI.3.1 Finding Smooth Twins with the CHM Algorithm

Conrey, Holmstrom, and McLaughlin [CHM13] present the following algorithm for producing many B-smooth values of $X(X+1)$. It starts with the initial set

$$S^{(0)} = \{1, 2, \ldots, B-1\}$$

of all integers less than B, representing the B-smooth twins $(1,2)$, $(2,3)$, ..., $(B-1, B)$. Next, it iteratively passes through all pairs of distinct $r, s \in S^{(0)}, r < s$ and computes

$$\frac{t}{t'} = \frac{r}{r+1} \cdot \frac{s+1}{s},$$

writing $\frac{t}{t'}$ in lowest terms. If $t' = t+1$, then clearly t also represents a twin smooth pair. The next set $S^{(1)}$ is formed as the union of $S^{(0)}$ and the set of all solutions t such that $t' = t+1$. Now the algorithm iterates through all pairs of distinct $r, s \in S^{(1)}$ to form $S^{(2)}$ and so on. We call the process of obtaining $S^{(d)}$ from $S^{(d-1)}$ the d-th CHM iteration. Once $S^{(d)} = S^{(d-1)}$, the algorithm terminates.

Example 1. *We illustrate the algorithm for $B = 5$, i.e. with the goal to generate 5-smooth twin integers. The starting set is*

$$S^{(0)} = \{1, 2, 3, 4\}.$$

Going through all pairs $(r,s) \in S^{(0)}$ with $r < s$, we see that the only ones that yield a new twin smooth pair $(t, t+1)$ via Equation (VI.1) *with t not already in $S^{(0)}$ are $(2,3)$, $(2,4)$ and $(3,4)$, namely,*

$$\frac{2}{2+1} \cdot \frac{3+1}{3} = \frac{8}{9}, \quad \frac{2}{2+1} \cdot \frac{4+1}{4} = \frac{5}{6}, \quad \textit{and} \quad \frac{3}{3+1} \cdot \frac{4+1}{4} = \frac{15}{16}.$$

Hence, we add 5, 8 *and* 15 *to get the next set as*

$$S^{(1)} = \{1, 2, 3, 4, 5, 8, 15\}.$$

The second and third CHM iterations give

$$S^{(2)} = \{1, 2, 3, 4, 5, 8, 9, 15, 24\} \text{ and } S^{(3)} = \{1, 2, 3, 4, 5, 8, 9, 15, 24, 80\}.$$

The fourth iteration does not produce any new numbers, i.e. we have $S^{(4)} = S^{(3)}$, the algorithm terminates here and returns $S^{(3)}$. This is indeed the full set of twin 5*-smooth integers as shown in [Stø97], see also [Leh64, Table 1A].*

Remark VI.5. *The CHM check that determines whether a pair (r, s) yields an integer solution t to the equation*

$$\frac{t}{t+1} = \frac{r}{r+1} \cdot \frac{s+1}{s} \tag{VI.1}$$

can be rephrased by solving this equation for t, which yields

$$t = \frac{r(s+1)}{s-r}. \tag{VI.2}$$

This shows that in order for (r, s) to yield a new pair, $s - r$ must divide $r(s + 1)$ and in particular, must be B-smooth as well.

VI.3.2 Generalising the CHM Algorithm

We now present a generalisation of the CHM algorithm, which finds smooth values of any monic quadratic polynomial $f(X) = X^2 + aX + b \in \mathbb{Z}[X] \subseteq \mathbb{Q}[X]$. The algorithm works with elements in the $\mathbb{Q}$-algebra $A = \mathbb{Q}[X]/\langle f(X)\rangle$. Let $\bar{X}$ denote the residue class of X in A. The generalisation closely follows the idea of the CHM algorithm and is based on the observation that for any $r \in \mathbb{Q}$, we have that

$$N_{A/\mathbb{Q}}(r - \bar{X}) = f(r),$$

where $N_{A/\mathbb{Q}}(\alpha)$ denotes the algebraic norm of $\alpha \in A$ over $\mathbb{Q}$. The algorithm now starts with an initial set

$$S^{(0)} = \{r_1 - \bar{X}, \ldots, r_d - \bar{X}\},$$

where r_i are smooth integer values of $f(X)$, which means that the element $r_i - \bar{X}$ has smooth non-zero norm. Next, in the d-th iteration of the algorithm, given any two $\alpha, \beta \in S^{(d-1)}$, compute

$$\alpha \cdot \beta^{-1} \cdot N_{A/\mathbb{Q}}(\beta) = r - s\bar{X}$$

for integers r, s (notice that β is invertible, since it has non-zero norm). Now, if s divides r, we obtain an integer $t = \frac{r}{s}$. It follows that

$$\begin{aligned} f(t) &= N_{A/\mathbb{Q}}\left(\frac{r}{s} - \bar{X}\right) \\ &= N_{A/\mathbb{Q}}(r - s\bar{X})s^{-2} \\ &= N_{A/\mathbb{Q}}(\alpha \cdot \beta^{-1} \cdot N_{A/\mathbb{Q}}(\beta))s^{-2} \\ &= N_{A/\mathbb{Q}}(\alpha) N_{A/\mathbb{Q}}(\beta) s^{-2}. \end{aligned}$$

Since both $N_{A/\mathbb{Q}}(\alpha)$ and $N_{A/\mathbb{Q}}(\beta)$ are B-smooth and s is an integer, it follows that t is a B-smooth value of $f(X)$. The set $S^{(d)}$ is then formed as the union of $S^{(d-1)}$ and the set of all such integral solutions. Finally, we terminate when $S^{(d)} = S^{(d-1)}$.

VI.3.3 Equivalence with Previous Algorithms

We now show that the CHM algorithm, as well as another algorithm by Conrey and Holmstrom [CH21], are special cases of the generalised algorithm, for the polynomials $f(x) = X^2 + X$, and $f(X) = X^2 + 1$ respectively.

Smooth values of $X^2 + X$. To see that the CHM algorithm (see Section VI.3.1) is indeed a special case of the generalised algorithm above, we show how the generalised algorithm works for $f(X) = X(X+1) = X^2 + X$. Consider the algebra $A = \mathbb{Q}[X]/\langle X^2 + X\rangle$. This embeds into the matrix algebra $M_{2\times 2}(\mathbb{Q})$ via

$$\psi : r + s\bar{X} \to \begin{pmatrix} r & 0 \\ s & r - s \end{pmatrix}.$$

Instead of working with elements in A, we will work with elements in $\psi(A) \subseteq M_{2\times 2}(\mathbb{Q})$ since this simplifies the argument. In this case, for $\alpha \in A$, we have

$$N_{A/\mathbb{Q}}(\alpha) = \det(\psi(\alpha)).$$

The set corresponding to the initial set in the CHM algorithm is

$$S^{(0)} = \{\left(\begin{smallmatrix} 1 & 0 \\ -1 & 2 \end{smallmatrix}\right), \left(\begin{smallmatrix} 2 & 0 \\ -1 & 3 \end{smallmatrix}\right), \ldots, \left(\begin{smallmatrix} B-1 & 0 \\ -1 & B \end{smallmatrix}\right)\}.$$

All these elements clearly have B-smooth norm. The d-th CHM iteration proceeds as follows: For all $\left(\begin{smallmatrix} r & 0 \\ -1 & r+1 \end{smallmatrix}\right), \left(\begin{smallmatrix} s & 0 \\ -1 & s+1 \end{smallmatrix}\right)$ in $S^{(d-1)}$, we try

$$\begin{aligned}\begin{pmatrix} r & 0 \\ -1 & r+1 \end{pmatrix}\begin{pmatrix} s & 0 \\ -1 & s+1 \end{pmatrix}^{-1} s(s+1) &= \begin{pmatrix} r & 0 \\ -1 & r+1 \end{pmatrix}\left(\begin{pmatrix} s+1 & 0 \\ 1 & s \end{pmatrix}\frac{1}{s(s+1)}\right)s(s+1) \\ &= \begin{pmatrix} r(s+1) & 0 \\ -(s-r) & (r+1)s \end{pmatrix}.\end{aligned}$$

Finally, we transform this matrix into the right form, i.e. into a matrix corresponding to an element of the form $\tau = t - \bar{X}$, which means that $\psi(\tau)$ has a -1 in the lower left corner. So, we divide by $s - r$ and end up with the matrix

$$\begin{pmatrix} \frac{r(s+1)}{s-r} & 0 \\ -1 & \frac{(r+1)s}{s-r} \end{pmatrix} = \begin{pmatrix} \frac{r(s+1)}{s-r} & 0 \\ -1 & \frac{r(s+1)}{s-r} + 1 \end{pmatrix}.$$

Now if $\frac{r(s+1)}{s-r}$ is an integer, we add this matrix to the next set $S^{(d+1)}$.

As we have seen in Remark VI.5, this integer indeed corresponds to the solution (VI.2) of Equation (VI.1) and therefore, the generalised algorithm in the case $f(X) = X^2 + X$ is equivalent to the original CHM algorithm.

Smooth values of $X^2 + 1$. Conrey and Holmstrom later presented a method to generate smooth values of $X^2 + 1$ [CH21]. Similar to the CHM algorithm, it starts with an initial set $S^{(0)}$ of smooth values of $X^2 + 1$. Again, for $d > 0$ and given $r, s \in S^{(d-1)}, r < s$, they compute

$$\frac{rs - 1}{s + r}.$$

The next set $S^{(d)}$ is then again formed as the union of $S^{(d-1)}$ and the set of all such values that are integers.

It is equally straightforward to verify that this algorithm is also a special case of the generalised CHM algorithm described above in Section VI.3.2. We could again work with matrices in $M_{2\times 2}(\mathbb{Q})$, but here, we are actually working in the number field $K = \mathbb{Q}[X]/\langle X^2+1\rangle$, which is isomorphic to $\mathbb{Q}(i)$, where $i^2 = -1$. The product of the elements $\alpha = r - i$ and $\beta = s - i$ is given as

$$\alpha\beta = (r-i)(s-i) = (rs-1) - (r+s)i.$$

Conrey and Holmstrom's method then simply tries all such products $\alpha\beta$. However, a possibly better choice could be to use

$$\alpha\beta^{-1}N_{K/\mathbb{Q}}(\beta) = \alpha\bar{\beta} = (r-i)(s+i) = (rs+1) - (s-r)i$$

as described in our generalisation. This is due to the fact that

$$\frac{rs+1}{s-r}$$

is more likely to be an integer, which means that we can expect the algorithm to converge faster.

Whichever option is chosen, one tries to divide by $r+s$ resp. $s-r$, and if the result is an element in $\mathbb{Z}[i]$, it is added to the next set $S^{(d)}$ of smooth values of X^2+1. Conrey and Holmstrom's method is therefore another special case of the generalised algorithm.

Remark VI.6. *We note that neither the generalised CHM algorithm, nor any of the previous special cases give any guarantees to what proportion of B-smooth values of $f(X)$ it finds. However, for the previous special case algorithms, certain conjectural results have been stated, based on numerical evidence, which suggests that the algorithm returns all but a small fraction of all smooth values of the respective quadratic polynomials. We make no similar claims for the general case algorithm.*

VI.4 Searching for Large Twin Smooth Instances: CHM in Practice

Ideally, the CHM algorithm could be run as described in [CHM13] with a large enough smoothness bound B to find twin smooths of cryptographic sizes. However, experiments suggest that this is not feasible in practice. We report on data obtained from an implementation of the pure CHM algorithm in Section VI.4.1, present several optimisations in Section VI.4.2 and details on our optimised implementation in Section VI.4.3.

VI.4.1 Running CHM in Practice

In order to collect data and assess the feasibility of finding large enough twin smooths, we implemented a somewhat optimised version of the pure CHM algorithm. In particular, this implementation is parallelised, and avoids multiple checks of the same pairs of twin smooths (r,s). Furthermore, we iterate through smoothness bounds: We start with a small bound B_1 and the initial set $S_1^{(0)} = \{1,\dots,B_1-1\}$, and use the CHM algorithm to iteratively compute sets $S_1^{(i)}$ until we reach some d_1 such that $S_1^{(d_1)} = S_1^{(d_1-1)}$. In the next iteration, we increase the smoothness bound to $B_2 > B_1$ and define the initial set $S_2^{(0)} = S_1^{(d_1)} \cup \{B_1,\dots,B_2-1\}$. Again we compute CHM iterations until we find d_2 such that $S_2^{(d_2)} = S_2^{(d_2-1)}$, where we avoid checking pairs (r,s) that have been processed in earlier iterations. Ideally, we could repeat this procedure until we reach a smoothness bound B_i for which the CHM algorithm produces large enough twin smooths for cryptographic purposes. However, our data suggests that this is infeasible in practice due to both runtime and memory limitations.

In particular, we ran this approach up to the smoothness bound $B = 547$, and extrapolating the results gives us rough estimations of the largest possible pair and number of twin smooths per smoothness bound.

After the $B = 547$ iteration, the set of twin smooths contains 82,026,426 pairs, whose bitlength distribution roughly resembles a normal distribution centered around bitlength 58. The largest pair has a bitlength of 122 bits. An evaluation of the obtained set is shown in Figure 10. Figure 10a shows the distribution of bitsizes in the full set, while Figure 10b shows that of the subset of all 199-smooth twins obtained in this run. Figure 10c shows the bitsize of the largest q-smooth twin pairs for each prime q between 3 and 547. And Figures 10d and 10e show the number of q-smooth twins for each such q.

Using the data of these experiments, we can attempt to estimate at which smoothness bound B this approach can be expected to reach twin smooths of cryptographic sizes, and how much memory is required to run iterations to reach this B. The data visualised in Figure 10c indicates that the bound necessary for the largest twin smooth pair obtained by running CHM with this bound to reach a bitlength of 256 lies in the thousands, possibly larger than 5,000. Similarly, the data displayed in Figures 10d and 10e shows how quickly the number of B-smooth twins increases with B. Given that the effort for CHM iterations grows quadratically with the set size, these estimates indicate that it is not feasible to reach cryptographically sized smooth twins with the original CHM algorithm.

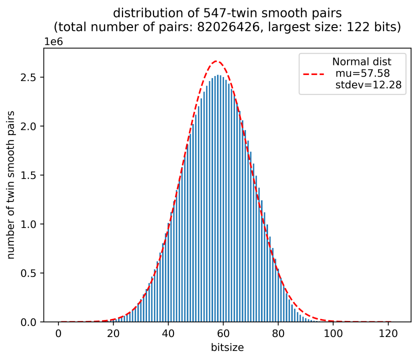
VI.4.2 Optimisations

One major issue with running the plain CHM algorithm for increasing smoothness bound is the sheer size of data that needs to be dealt with. The sets $S_i^{(d_i)}$ grow very rapidly and the quadratic complexity of checking all possible pairs (r,s) leads to a large runtime. The natural question that arises is whether CHM can be restricted to checking only a certain subset of such pairs without losing any or too many of the new smooth twins. Furthermore, if the purpose of running the CHM algorithm is not to enumerate all twin smooth pairs for a given smoothness bound but instead, to produce a certain number of pairs of a given size or to obtain some of the largest pairs, it might even be permissible to omit a fraction of pairs.

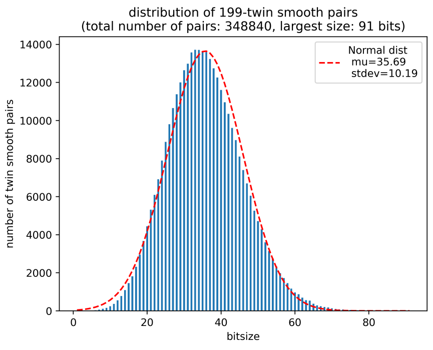
To find a sensible way to restrict to a smaller set, we next discuss which pairs (r,s), $r < s$ result in a given twin smooth pair $(t, t+1)$ via

$$\frac{r}{r+1} \cdot \frac{s+1}{s} = \frac{t}{t+1}. \tag{VI.3}$$

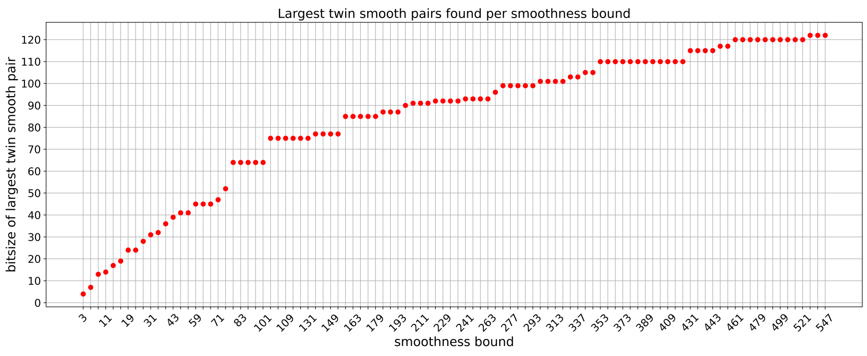
This is discussed in [CHM13, Section 3], but we elaborate on it in a slightly different way here. Let $t > 0$, let u be any divisor of t and v any divisor of $t+1$. Let $h, x \in \mathbb{Z}$ be given by $t = uh$ and $t+1 = vx$ (where $u, v, h, x > 0$). Therefore, $v/u = h/x + 1/(ux)$. If $u < v$ then $h > x$ and if $u > v$ then $h < x$. We therefore fix $u < v$ (otherwise switch the roles of u, v and h, x). Since



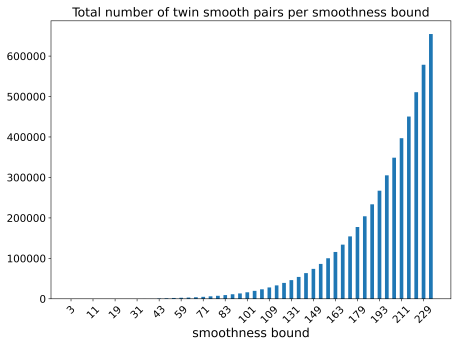
(a) Distribution of bitsizes for the full set of 547-*twin smooth pairs.*



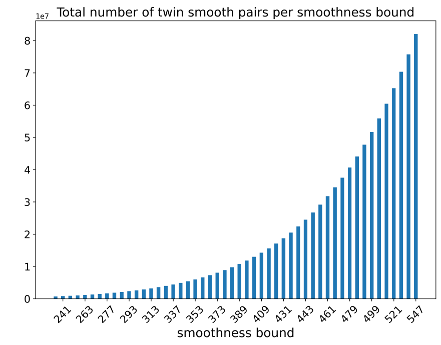
(b) Distribution of bitsizes for the subset of 199-*twin smooth pairs.*



(c) Bitsizes of the largest q-smooth twins for all primes q between 3 *and* 547.



(d) Number of q-smooth twins for all primes q between 3 *and* 233.



(e) Number of q-smooth twins for all primes q between 239 *and* 547.

Figure 10: Evaluation of the set of 547*-smooth twins obtained by running the original CHM algorithm with smoothness bound $B = 547$. The bitsize of a pair $(r, r+1)$ is $\lfloor \log r \rfloor + 1$. Data for the number of q-smooth twins for all primes q up to* 547 *has been split into two histograms of different scale.*

$u < v$, the pair

$$(r,s) = (t - \frac{u}{v}(t+1),\ \frac{v}{u}t - (t+1) = \frac{v}{u}r) \tag{VI.4}$$

satisfies Equation (VI.3) and it follows that

$$r = u(h-x),\ r+1 = x(v-u),\ s = v(h-x),\ s+1 = h(v-u). \tag{VI.5}$$

Therefore, $s/r = v/u$ and $(s+1)/(r+1) = h/x$, $u < v$, $h > x$ and $0 < r < s$. This also means that $s = r + (v-u)(h-x)$, $t = r + ux$ and that $\gcd(r(s+1), s(r+1)) = s - r = (v-u)(h-x)$ (note that $\gcd(uh, vx) = \gcd(t, t+1) = 1$).

Conversely, given (r,s) with $r > 0$ that satisfy Equation VI.3, define $u = r/\gcd(r,s)$ and $v = s/\gcd(r,s)$, then $s > r$, $u < v$ and $u \mid t$, $v \mid (t+1)$. Hence we have the correspondence between the set of pairs (r,s) with $r < s$ that yield a new twin pair $(t, t+1)$ via Equation (VI.3) and the set of pairs of divisors of t and $t+1$ described in [CHM13, Section 3] as follows:

$$\begin{aligned}&\{(r,s) \mid r < s \text{ and } r(s+1)(t+1) = s(r+1)t\}\\ &\qquad \longleftrightarrow \{(u,v) \mid u < v \text{ and } u \mid t,\ v \mid (t+1)\}.\end{aligned} \tag{VI.6}$$

However, this correspondence does not identify the pairs (r,s) corresponding to twin smooths, i.e. given (u,v) there is no guarantee that any of $r, r+1, s, s+1$ are B-smooth. This is not discussed in [CHM13, Section 3]. The next lemma fills this gap by stating an explicit condition on the divisors u, v, h, x.

Lemma VI.7. *Let $t \in \mathbb{Z}$ such that $t(t+1)$ is B-smooth. Let (u,v) be a pair of divisors such that $t = uh$, $t+1 = vx$ and let (r,s) be defined as in Equation* (VI.4).

Then $r(r+1)s(s+1)$ is B-smooth if and only if $(v-u)(h-x) = s - r$ is B-smooth.

Proof. As divisors of t and $t+1$, u and v as well as h and x are all B-smooth. The statement follows from the Equations (VI.5). □

Using similar sized pairs. We next consider the following condition to restrict the visited pairs (r,s) in CHM as a mechanism to reduce the set size and runtime. Let $k > 1$ be a constant parameter. We then only check pairs (r,s) if they satisfy

$$0 < r < s < kr. \tag{VI.7}$$

Assume that (r,s) results in a pair $(t, t+1)$ through satisfying Equation (VI.3). As seen above, $\frac{s}{r} = \frac{v}{u}$ for $u \mid t$, $v \mid (t+1)$, so we can use (u,v) to determine which values k are useful. Since $\frac{v}{u} < k$, it follows $s = \frac{v}{u}t - (t+1) < (k-1)t$. If we are only interested in obtaining a new t from a pair (r,s) such that $s < t$, we can take $k \leq 2$, overall resulting in $1 < k \leq 2$.

This k seems to be a good quantity to study as we can relate it to the factors of $v - u$. Indeed, $v - u = u(\frac{v}{u} - 1) = u(\frac{s}{r} - 1)$ and we have $s < kr$.

Definition VI.8. *Let $(r, r+1)$ and $(s, s+1)$ be twin smooths with $r < s$ and $k \in \mathbb{R}$ with $1 < k \leq 2$. We call the pair (r,s) k-*balanced *if $r < s < k \cdot r$.*

We want to find a k such that a k-balanced pair (u, v) subject to the above conditions will yield a balanced r, s such that $r, r+1, s, s+1$ are B-smooth, or equivalently that $v-u$ and $h-x$ are.

Running the CHM algorithm only with 2-balanced pairs (r, s) then guarantees that any t produced by Equation VI.3 will be larger than the inputs r and s. Although we sacrifice completeness of the set of twin B-smooths with this approach, we can significantly reduce the runtime.

We can even push this approach further. Recall that we require $\gcd(r(s+1),(r+1)s) = s-r$ in order to generate a new pair of twin smooths $(t, t+1)$. By Lemma VI.7, this can only hold if $\Delta = s-r$ is B-smooth. Hence, only checking pairs (r, s) for which Δ is likely to be smooth increases the probability for a successful CHM step. Heuristically, the smaller Δ is, the better the chances for Δ to be smooth. Furthermore, if Δ contains small and only few prime factors, the probability for the condition $\Delta = \gcd(r(s+1),(r+1)s)$ is relatively high. We can summarise this in the following heuristic.

Heuristic VI.9. *Let $k_1, k_2 \in \mathbb{R}$ with $1 < k_1 < k_2 \leq 2$, and (r_1, s_1) resp. (r_2, s_2) a k_1- resp. k_2-balanced pair of twin smooths. Then the probability for (r_1, s_1) to generate new twin smooths via the CHM equation is larger than that for (r_2, s_2).*

In order to save additional runtime, we can thus pick k closer to 1, and only check the pairs (r, s) that are most likely to generate new twin smooths. Therefore, we can still expect to find a significant portion of all twin B-smooths for a given smoothness bound B. We expand on the choice of k and different ways of implementing this approach in Section VI.4.3.

Thinning out between iterations. Another approach to reduce both runtime and memory requirement is to thin out the set of twin smooths between iterations. In particular, once we finished all CHM steps for a certain smoothness bound B_i, we can remove twins from the set $S_i^{(d_i)}$ based on their likeliness to produce new twin smooths before moving to the next iteration for B_{i+1}.

One possible condition for removing twins is to look at their smoothness bounds. Let $(r, r+1)$ be B_1-smooth, $(s, s+1)$ be B_2-smooth (but not B-smooth for any $B < B_2$), and $B_1 \ll B_2$. Since $(s, s+1)$ contains (multiple) prime factors larger than B_1, they cannot be contained in $(r, r+1)$, which makes the requirement $\gcd(r(s+1),(r+1)s) = s-r$ heuristically less likely to be satisfied. However, in practice it turns out that the differences between the smoothness bounds we are concerned with are not large enough for this heuristic to become effective.

In our experiments, it turned out to be more successful to keep track of how many new twin smooths each r produces. We can then fix some bound m, and discard twins that produced less then m twins after a certain number of iterations. Our experiments suggest that using this approach with carefully chosen parameters yields a noticeable speedup, but fails completely at reducing the memory requirements, as we still need to keep track of the twins we already found. Furthermore, in practice the approach of only using k-balanced twins turned out to be superior, and hence we focus on this optimisation in the following.

VI.4.3 Implementation

We implemented the CHM algorithm with several of the aforementioned optimisations in C++, exploiting the fact that it parallelises perfectly. Note that some of our approaches require the set of twin smooths to be sorted with respect to their size. Hence, an ordered data structure is used for storing the twins set. We used the following techniques and optimisations.

CHM step. For each pair (r, s) considered by the implementation, we have to check if Equation (VI.3) holds. As mentioned in Section VI.4.2, this requires that $\gcd(r(s+1), (r+1)s) = s-r$ is satisfied. However, we can completely avoid the gcd calculation by observing that we require $r \cdot (s+1) \equiv 0 \mod (s-r)$. Only if this is the case we perform a division to compute t, which represents the new pair of twin smooths $(t, t+1)$. Therefore, we only perform one modular reduction per considered pair (r, s), followed by one division if the CHM step is successful. This is significantly cheaper than a naïve implementation of Equation (VI.3) or a gcd computation.

Data structure. Initially the set of twins was organised in a standard C array, that each time an iteration completed was reallocated to increase its size, and reordered.

To avoid the overall inefficiency of this method we moved to use the C++ standard library std::set. This data structure is implemented with a Red Black tree, guarantees $O(\log N)$ insertion and search, while keeping the elements always ordered.

We then moved to use B+Trees [Bin18], that have the same guarantees for insertion, search, and ordering, but are more efficient in the memory usage. Because the elements of a B+Tree are stored close to each other in memory it becomes much faster to iterate through the set, an operation that is necessary for creating the pairs used in each computation.

Implemented optimisations. As discussed in Section VI.4.2, we focus on the case of k-balanced pairs (r, s), which satisfy $r < s < k \cdot r$. Compared to the full CHM algorithm, this leads to a smaller set of twin smooths, but allows for much faster running times. We implemented the k-balanced approach in various different flavours.

Global-k: In the simplest version - the `global-`k approach - we initially pick some k with $1 < k \leq 2$, and restrict the CHM algorithm to only check k-balanced pairs (r, s). The choice of k is a subtle manner: Picking k too close to 1 may lead to too many missed twin smooths, such that we cannot produce any meaningful results. On the other hand, picking k close to 2 may result in a relatively small speedup, which does not allow for running CHM for large enough smoothness bounds B. Unfortunately, there seems to be no theoretical handle on the optimal choice of k, which means that it has to be determined experimentally. We note that when picking an aggressive bound factor $k \approx 1$, small numbers r in the set of twins S may not have any suitable $s \in S$ they can be checked with. Thus, we pick a different bound, e.g. $k = 2$, for numbers below a certain bound, e.g. for $r \leq 2^{20}$.

Iterative-k: Instead of iterating through smoothness bounds B_i as described in Section VI.4.1 and using the `global-`k approach, we can switch the roles of B and k if we are interested in running CHM for a fixed smoothness bound B. We define some initial value k_0, a target value $k_{\max}$, and a step size $k_{\text{step}} > 0$. In the first iteration, we run CHM as in the `global-`k approach, using k_0. The next iteration then increases to $k_1 = k_0 + k_{\text{step}}$, and we add the condition to not check pairs (r,s) if they were already checked in previous iterations. We repeat this iteration step several times until we reach $k_{\max}$. Compared to the `global-`k approach, this allows us to generate larger B-smooth twins faster, since we restrict to the pairs (r,s) first that are most likely to generate new twins. However, the additional checks if previous pairs have been processed in earlier iterations add a significant runtime overhead. Thus, this method is more suitable for finding well-suited choices of k, while actual CHM searches benefit from switching to the `global-`k approach.

Constant-range: In both the `global-`k and `iterative-`k approach, the checks if a pair (r,s) is k-balanced, or has been processed in earlier iterations, consumes a significant part of the overall runtime. Therefore, we can use constant ranges to completely avoid these checks. Since we always keep the set of twins S sorted by size, the numbers s closest to r (with $s > r$) are its neighbours in S. Thus, we can sacrifice the exactness of the k-balanced approaches above, and instead fix a range R and for each r check (r,s) with the R successors s of r in S. As shown below, this method significantly outperforms the `global-`k approach due to the elimination of all checks for k-balance. This is true even when R is large enough to check more pairs than are considered in the `global-`k approach for a given k.

Variable-range: Similar to the `constant-range` approach, we can adapt the range R depending on the size of r. For instance, choosing r at the peak of the size distribution will lead to many possible choices of s such that (r,s) are balanced. Hence, we can choose a larger range R whenever more potential pairs exist, while decreasing R otherwise. In practice, the performance of this method ranks between `global-`k and `constant-range` by creating roughly the same pairs that `global-`k creates without any of the overhead of the balance checks. If R is chosen large enough such that the `constant-range` approach ends up generating more pairs than `global-`k, then `variable-range` performs better. Realistically, the size of the range R increases by (very) roughly 3% for each prime number smaller than the smoothness bound B, and slows down the algorithm drastically at higher smoothness, similarly to the k-based approaches.

Remark VI.10. *Similar to the* `variable-range` *approach, we experimented with a variant of the* `global-`k *approach, which adjusts k according to the size of r to find suitable s for the CHM step. However, the* `constant-range` *and* `variable-range` *approaches turned out to be superior in terms of performance, and therefore we discarded this* `variable-`k *variant.*

Performance comparison. In order to compare the implications of the optimisations in practice, we ran different variants of the CHM implementation for the fixed smoothness bound $B = 300$. All experiments ran on a machine configured with 4 x Xeon E7-4870v2 15C 2.3 GHz, 3072 GB of RAM. The total amount of parallel threads available was 120. As described above, the `global-`k and `constant-range` approach significantly outperform their respective variants, hence we focus on different configurations of these two methods.

The results are summarised in Table 1. For both the `global-`k and the `constant-range` approach we measured the results for conservative and more aggressive instantiations, where smaller values of k and R are considered more aggressive. It is evident that already for the conservative instantiations, we gain significant performance speedup, while still finding almost

Variant	Parameter	Runtime	Speedup	#twins	#twins from largest 100
Full CHM	-	4705s	1	2300724	100
`global-`k	$k = 2.0$	364s	13	2289000	86
	$k = 1.5$	226s	21	2282741	82
	$k = 1.05$	27s	174	2206656	65
`constant-range`	$R = 10000$	82s	57	2273197	93
	$R = 5000$	35s	134	2247121	87
	$R = 1000$	16s	294	2074530	75

Table 1: Performance results for different variants of our CHM implementation for smoothness bound $B = 300$. Speedup factors refer to the full CHM variant.

the full set of twin smooths, and most of the 100 largest 300-smooth twins. For the more aggressive instantiations, we miss more twins, yet still find a significant amount of large twins.

As discussed above, the `constant-range` approach outperforms the `global-`k approach in terms of runtime, due to the elimination of all checks for k-balance of twins. Interestingly, while very aggressive instantiations of `constant-range` miss more twin smooths, they find a larger share of the largest 100 twins than their `global-`k counterpart. Therefore, we conclude that for larger smoothness bounds B, for which we cannot hope to complete the full CHM algorithm, `constant-range` is the most promising approach for obtaining larger twin smooths within feasible runtimes.

Remark VI.11. *While all optimisations lose a small proportion of the largest twin smooths, they are not necessarily lost permanently. In practice, when iterating to larger smoothness bounds B_i, we often also find some B_j-smooth twins for bounds $B_j < B_i$. Thus, the size of the set of* 300*-smooth twins usually increases in the optimised variants when moving to larger B.*

Remark VI.12. *In the following sections, we will require twin smooths of a certain (relatively small) bitlength. This can easily be incorporated into all implemented variants by removing all twins above this bound after each iteration. This means that we cut off the algorithm at this size, and do not attempt to obtain larger twins, which significantly improves the runtime and memory requirements.*

VI.4.4 Record Twin Smooth Computations

We ran the optimised full CHM algorithm with $B = 547$ and found a total of 82,026,426 pairs of B-smooth twins. Among these pairs, we found 2,649 additional 200-smooth twins that were not found by the original authors of the algorithm [CHM13]. This showcases the validity of Remark VI.6 that the algorithm does not guarantee us to find all B-smooth twins. Furthermore, there is no guarantee that running CHM with $B = 547$ will produce all 200-smooth twins. As mentioned in the introduction, the only way to see how far away we are from the exact number of 200-smooth twins is to solve all 2^{46} Pell equations.

For the application mentioned in the upcoming sections within this chapter, we only need twins of a certain bitsize. Within this set of twins, 9,218,648 pairs $(r, r + 1)$ fall in the range $2^{60} < r < 2^{64}$; 1,064,249 pairs fall in the range $2^{81} < r < 2^{85}$; 31,994 pairs fall in the range $2^{92} < r < 2^{96}$; and, only 1 pair falls in the range $2^{120} < r < 2^{128}$. This pair in the final interval is the largest pair found in this run, with $r = 40120312418488665264241657960477 49375$, and factorisations:

$$\begin{aligned} r &= 5^4 \cdot 7 \cdot 13^2 \cdot 17^2 \cdot 19 \cdot 29 \cdot 41 \cdot 109 \cdot 163 \cdot 173 \cdot 239 \cdot 241^2 \cdot 271 \cdot 283 \\ &\quad \cdot 499 \cdot 509, \text{ and} \\ r+1 &= 2^8 \cdot 3^2 \cdot 31^2 \cdot 43^2 \cdot 47^2 \cdot 83^2 \cdot 103^2 \cdot 311^2 \cdot 479^2 \cdot 523^2. \end{aligned}$$

As we will see later, the number of 64-bit and 85-bit twins we found in this run is enough to find attractive parameters for SQISign. The 96-bit twins will give us parameters with the required smoothness, however we do not have enough pairs to hope to find a prime p where $p^2 - 1$ is divisible by a large power of two.

As mentioned earlier, increasing the smoothness bound B and running the full CHM algorithm cannot hope to terminate in some reasonable amount of time. However, using the optimisations introduced in the VI.4.3, we can increase B and expect the algorithm to terminate and hopefully find larger sized twins. So we ran the algorithm for $B = 1300$ using the `constant-range` optimisation with a range $R = 5000$, in order to specifically target twins $(r, r+1)$ with $r > 2^{115}$. In this run we found 1,091 such pairs - the largest of these pairs is the following 145-bit twin $(r, r+1)$ with $r = 36132012096025817587153962195378848686084640$, where

$$\begin{aligned} r &= 2^5 \cdot 5 \cdot 7 \cdot 11^2 \cdot 13 \cdot 23 \cdot 53 \cdot 71 \cdot 109 \cdot 127 \cdot 131 \cdot 193 \cdot 251 \cdot 283 \cdot 307 \\ &\quad \cdot 359 \cdot 367 \cdot 461 \cdot 613 \cdot 653 \cdot 1277, \text{ and} \\ r+1 &= 3^2 \cdot 29^2 \cdot 31^2 \cdot 43^2 \cdot 59^2 \cdot 61^2 \cdot 73^2 \cdot 79^2 \cdot 89^2 \cdot 167^2 \cdot 401^2 \cdot 419^2. \end{aligned}$$

Among the 1,091 twins CHM found, 184 pairs fall in the range $2^{120} < r < 2^{128}$, which was sufficient to find some SQISign-friendly parameters (though not at all NIST security levels).

In addition, we also ran CHM with $B = 2^{11}$ to obtain a large number of twin smooth integers in the range $2^{55} < r < 2^{100}$ (see Remark VI.12 in the setting where we want to find twins in such an interval). This run was performed using the `constant-range` optimisation with a range $R = 2500$, and produced 608,233,761 pairs of twins lying in this range. Compared with the $B = 547$ run, the yield from this run gave ample twins with $2^{92} < r < 2^{96}$, which was sufficient to find SQISign parameters with the desirable large power of two.

VI.5 Cryptographic Primes of the form $p = 2r^n - 1$: Powerlift CHM twins

This section focuses on finding primes suitable for isogeny-based cryptographic applications. As discussed in the previous sections, the pure CHM method does not allow for us to directly compute twins of at least 256 bits as required for this aim. However, as mentioned in Chapter IV through the description isogeny-based signature scheme SQISign, some cryptographic applications do not need twins $(r, r+1)$ that are fully smooth. As also mentioned in Chapter IV, the current incarnation of SQISign requires a prime p that satisfies $2^f T \mid p^2 - 1$, where f is as large as possible, and $T \approx p^{5/4+\epsilon}$ is smooth and odd [DFLLW23]. This flexibility allows us to move away from solely using CHM and, instead, to use CHM results as inputs to known methods for

n	$p_n(x)^2 - 1$
2	$4x^2(x-1)(x+1)$
3	$4x^3(x-1)(x^2+x+1)$
4	$4x^4(x-1)(x+1)(x^2+1)$
5	$4x^5(x-1)(x^4+x^3+x^2+x+1)$
6	$4x^6(x-1)(x+1)(x^2-x+1)(x^2+x+1)$

Table 2: Factorisation of $p_n(x)^2 - 1$ for $n = 2,3,4,5,6$, where $p_n(x) = 2x^n - 1$

finding such primes. At a high level, we will find fully smooth twins of a smaller bit-size via CHM and boost them up using the polynomials $p_n(x) = 2x^n - 1$ (for carefully chosen n). Hence, if $r, r+1$ are fully smooth integers and n is not too large, we can guarantee a large proportion of $p_n(r)^2 - 1$ to be smooth. For the sake of notation, we will often denote the evaluation of the polynomial $p_n(x)$ at some input r simply by $p = p_n(r)$ - emphasising that it is an integer. As a quick note, towards the end of this chapter, we utilise this boosting idea in a very similar manner but using other polynomials. As a result much of the theory that is described here can be translated to the latter section.

General method. We begin with a more in-depth description of the approach to obtaining cryptographic sized primes p, such that $p^2 - 1$ has $\log T'$ bits of B-smoothness, where $T' = 2^f T$ and B is some fixed smoothness bound. We recall that for our SQISign application, we have $\log p \in \{256, 384, 512\}$ for NIST Level I, III and V (respectively), $T \approx p^{5/4}$ and f as large as possible. In the current implementation of SQISign, $f \approx \lfloor \log(p^{1/4}) \rfloor$ (i.e., $T' \approx p^{3/2}$), and therefore, we aim for this when finding primes.

For the polynomials $p_n(x)$, we have $p_n(x)^2 - 1 = 4x^n(x-1)f(x)$ for some polynomial $f(x)$, as shown in Table 2. We observe that for n even, both $x+1$ and $x-1$ appear in the factorisation of $p_n(x)^2 - 1$. In this case, for twin smooths $(r, r \pm 1)$, evaluating $p_n(x)$ at r guarantees that we have a smooth factor $4x^n(x \pm 1)$ in $p^2 - 1$. For n odd, we will only have that $x - 1$ appears in the factorisation, and therefore only consider twins $(r, r-1)$ to guarantee we have B-smooth factor $4x^n(x-1)$.

The first step is to use our implementation of the CHM algorithm, described in Sections VI.3 and VI.4, to obtain B-smooth twins $(r, r \pm 1)$ of bitsize approximately $(\log p - 1)/n$. We then obtain primes of suitable sizes via computing $p = p_n(r)$ for all candidate r, as described above. By construction, $p^2 - 1$ has guaranteed $\frac{n+1}{n}(\log(p) - 1) + 2$ bits of smoothness. We then require that the remaining factors have at least

$$\max\left(0,\ \frac{3}{2}\log p - \left(\frac{n+1}{n}(\log p - 1) + 2\right)\right)$$

bits of B-smoothness. In Section VI.5.2, we will discuss the probability obtaining this smoothness from the remaining factors.

VI.5.1 Choosing n

For small n, we require CHM to find twin smooths of *large* bit size. For certain bit sizes, running full CHM may be computationally out of reach, and therefore we use a variant that may not find all twins. In this case, however, we have more guaranteed smoothness in p^2-1 and so it is more likely that the remaining factors will have the required smoothness. For large n, we can obtain more twin smooths from CHM (in some cases, we can even exhaustively search for all twin smooths), however we have less guaranteed smoothness in p^2-1. Finding values of n that balance these two factors will be the focus of this section.

n = 2. Let $(r, r\pm 1)$ be twin smooth integers and let $p = 2r^2-1$. In this case, $2r^2(r\pm 1) \mid T'$, meaning that $\log T' \geq \frac{3}{2}\log p$, and we have all the required smoothness. Write $T' = 2^f T = 2r^2(r\pm 1)$ where T is odd. If $f < \lfloor \log\left(p^{1/4}\right)\rfloor$, we have $T > p^{5/4}$, and we do not have to rely on a large power of 2 dividing $r-1$. Otherwise, we turn to Section VI.5.2 to estimate the probability of $r \mp 1$ having enough small factors to make up for this difference.

Suppose we target primes with λ bits of classical security, i.e., we need a prime of order $p \approx 2^{2\lambda}$. For $n = 2$, this corresponds to finding twin smooths of size $\approx 2^{\lambda-\frac{1}{2}}$, and so is only suitable for finding NIST Level I parameters due to the limitations of the CHM method (see Section VI.4). One could instead use other techniques for finding large enough twins for $n = 2$, such as the PTE sieve [CMN21], at the cost of significantly larger smoothness bounds. Alternatively, we can move to higher n, which comes at the cost of loosing guaranteed smoothness. Another challenge here is that, given the relatively large size of the twins, it appears difficult to find enough twins for obtaining primes with a large power of two.

n = 3. Let $(r, r-1)$ be twin smooth integers and let $p = 2r^3-1$. Here, we can guarantee that the smooth factor T' of p^2-1 is at least of size $\approx p^{4/3}$. If $f < \lfloor \log_2\left(p^{1/12}\right)\rfloor$, we have $T > p^{5/4}$. Otherwise, we require that there are enough smooth factors in r^2+r+1 to reach this requirement.

Here, for λ bits of classical security, we need to target twin smooth integers of size $\approx 2^{\frac{2\lambda-1}{3}}$. In this case, the CHM method will (heuristically) allow us to reach both NIST Level I and III parameters.

n = 4. Let $(r, r\pm 1)$ be twin smooth integers and $p = 2r^4-1$. Here we can only guarantee a factor of size $\approx p^{5/4}$ of p^2-1 to be smooth. When accounting for the power of two, we must hope for other smooth factors. As $p_n(x)-1$ splits into (relatively) small degree factors, namely $p_n(x)-1 = 2(x-1)(x+1)(x^2+1)$, the probability of having enough B-smooth factors is greater (than if there was, for example, a cubic factor).

In contrast to the previous cases, this setting should be suitable for targeting all necessary security parameters. However, for the NIST Level I setting, the work by De Feo, Leroux and Wesolowski [DFLLW23][Section 5.2] showed that the best one could hope for here while maximising the power of two gives SQISign parameters with a smoothness bound of ≈ 1800. While this is a better smoothness bound than the NIST Level I prime with the best performance for SQISign, it does not perform as well in practice. Indeed, most of the odd primes less than 1800 that appear in p^2-1 are relatively large, making isogeny computation relatively slow. In the best performing prime, however, a large power of 3 divides p^2-1, and most of its other odd prime divisors are fairly small. We note that the authors of [DFLLW23] only searched for parameters that maximise the power of two, and hence there could be some scope to find parameters that have slightly smaller powers of two.

Other n. For larger n, the amount of guaranteed smoothness decreases, and thus the probability that the remaining factors have the required smoothness is small. Indeed, we find that only $n = 6$ has the correct balance of requiring small twin smooths while still having a reasonable probability of success. This is primarily due to the factorisation of $p_6(x) - 1 = 2(x-1)(x+1)(x^2-x+1)(x^2+x+1)$, having factors of degree at most 2, which improves the probability that we have enough smooth factors. In contrast, $n = 5$ results in more guaranteed smoothness than $n = 6$, but requires the quartic factor in $p_5(x) - 1$ to provide the necessary smoothness, which is relatively unlikely.

While one could use $n = 6$ to find NIST Level I parameters, this larger n shines in its ability to give us both NIST Level III and V parameters.

VI.5.2 Probability of Sufficient Smoothness

In this section, we determine the probability of obtaining cryptographic primes with sufficient smoothness using the methods outlined above.

First, we determine the probability of finding fully B-smooth twins $(r, r \pm 1)$ using the counting function Ψ, defined in Section III.5.

To calculate the probability of p^2-1 having $\log T'$-bits of B-smoothness, given that the factor $r(r \pm 1) \mid p^2 - 1$ is already fully smooth, we will use the counting function Θ studied by Banks–Shparlinski and introduced in Section III.5. Indeed, as we only require around $\log T'$-bits of smoothness from $p^2 - 1$, it may only be partially smooth. As discussed in the section above, we restrict to $n = 2, 3, 4, 6$.

Recall that $p_n(x)^2 - 1 = 4x^n(x-1)f(x)$, where $f(x)$ is given Table 2 for each $2 \leq n \leq 6$. Write $f(x) = f_1(x) \cdots f_k(x)$, where each f_i is irreducible of degree $d_i = \deg(f_i)$ and $d = \deg(f)$. To calculate the probabilities, we require that the probability of $f(x)$ having at least $\log_2 D$-bits of B-smoothness is the product of the probabilities of each of its factors f_i having at least $\log_2 D_i$-bits of B-smoothness where $\log_2 D = \sum_{i=1}^{k} \log_2 D_i$. We can view this as an extension of [CMN21, Heuristic 1]. Note that the only constraint on how the smoothness is distributed between the factors $f_i(x)$ is that the total bit size of B-smooth factors must equal $\log_2 D$. We could, for example, sum over all the possible distributions of smoothness using the inclusion-exclusion principle. However, in distributions where one of the factors has a very small amount of smoothness, we fall out of the ranges allowed as input into Θ determined by Theorem III.8. Therefore, for simplicity, we will assume that smoothness is distributed evenly between the remaining factors (weighted by the degree), i.e., $\log_2 D_i = (d_i \log_2 D)/d$. In reality, this only gives us a lower bound for the probability, but this will suffice for our purposes. Obtaining a more theoretical and accurate grasp on these probabilities is left as an avenue for future research.

In Table 3, we give an overview of the relevant probabilities for NIST Level I, III, and V parameters, calculated as described above. We observe that as n gets larger, the probability of finding B-smooth integers of the appropriate bitsize increases. In contrast, for bigger n we are guaranteed less smoothness in $p^2 - 1$. As a result, given B-smooth twins, the probability of finding a SQISign prime p decreases as n increases. For each NIST level, we predict that the n

	n	$\log_2(r)$	**Probability of B-smooth** $(r, r \pm 1)$	**Probability of p^2-1 $\log T'$-bits B-smooth given $(r, r \pm 1)$ twin smooth**	**Extra Smoothness Needed**
NIST-I	2	≈ 127.5	$2^{-58.5}$	1	0
$B = 2^9$	3	≈ 85.0	$2^{-32.1}$	$2^{-12.1}$	42
$\log p = 256$	4	≈ 63.75	$2^{-20.5}$	$\approx 2^{-22.4}$	63.25
$\log T' = 384$	6	≈ 42.5	$2^{-10.4}$	$\approx 2^{-32.2}$	84.5
NIST-III	2	≈ 191.5	$2^{-55.7}$	1	0
$B = 2^{14}$	3	≈ 127.67	$2^{-30.5}$	$2^{-11.7}$	63.33
$\log p = 384$	4	≈ 95.75	$2^{-19.4}$	$\approx 2^{-15.7}$	95.25
$\log T' = 576$	6	≈ 63.83	$2^{-9.7}$	$\approx 2^{-19.2}$	127.17
NIST-V	2	≈ 255.5	$2^{-63.7}$	1	0
$B = 2^{17}$	3	≈ 170.33	$2^{-35.2}$	$2^{-13.5}$	84.67
$\log p = 512$	4	≈ 127.75	$2^{-22.6}$	$\approx 2^{-18.2}$	127.25
$\log T' = 768$	6	≈ 85.17	$2^{-11.5}$	$\approx 2^{-22.5}$	169.83

Table 3: Assuming that $(r, r \pm 1)$ are twin smooth integers and p has $\log p$ bits, calculates the probability of having a B-smooth divisor $T' \mid p^2 - 1$ of size $\approx p^{3/2}$. More details in text.

that balance these two contrasting probabilities have a higher chance of finding a p satisfying our requirements. As discussed in the next section, this trend is reflected in practice.

VI.5.3 Concrete Parameters for SQISign: Results and Comparisons

We now turn to giving a list of SQISign-friendly primes that target NIST Level I, III, and V. As mentioned previously, this means that we need to find primes p with $2^f \cdot T \mid p^2 - 1$. We need the exponent f to be as large as possible and the cofactor $T \approx p^{5/4+\epsilon}$ to be B-smooth, aiming to keep the ratio $\sqrt{B}/f$ as small as possible.

We find parameters for NIST Level I, III and V by searching for 256, 384 and 512-bit primes, respectively. For those primes targeting the higher security levels, these are the first credible SQISign-friendly primes. In what follows, we look at each security level and analyse the most noteworthy primes found in our searches. When stating the factorisations of $p \pm 1$ for the mentioned primes, the factors in grey are the rough factors which are not needed for SQISign. A full collection of our best SQISign-friendly primes that were found using the CHM machinery is showcased in Table 4.

NIST-I parameters. We targeted 256-bit primes using $n = 2, 3$ and 4. Given that our CHM runs produced a lot more twins of smaller bit-size compared to the 128-bit level, we expect to find more primes using $n = 3, 4$, which was indeed the case. It is worth noting that some primes found with $n = 2$ gave rise to $p^2 - 1$ being divisible by a relatively large power of two. However, in these cases, most of the primes dividing $p^2 - 1$ are relatively large and would therefore give rise to slower isogeny computations during the SQISign protocol [DFLLW23].

Through the experimentation with the 85-bit twins produced from CHM with $B = 547$, we found the following 254-bit prime $p = 2r^3 - 1$ with $r = 20461449125500374748856320$. All the specific criteria that we need for a SQISign parameter set are met, while obtaining an attractively small signing cost metric $\sqrt{B}/f$. For this prime, we have

$$\begin{aligned}
p+1 &= 2^{46} \cdot 5^3 \cdot 13^3 \cdot 31^3 \cdot 73^3 \cdot 83^3 \cdot 103^3 \cdot 107^3 \cdot 137^3 \cdot 239^3 \cdot 271^3 \cdot 523^3, \text{ and} \\
p-1 &= 2 \cdot 3^3 \cdot 7 \cdot 11^2 \cdot 17^2 \cdot 19 \cdot 101 \cdot 127 \cdot 149 \cdot 157 \cdot 167 \cdot 173 \cdot 199 \cdot 229 \cdot 337 \\
&\quad \cdot 457 \cdot 479 \cdot 141067 \cdot 3428098456843 \cdot 4840475945318614791658621.
\end{aligned}$$

While the associated cofactor T here exceeds $p^{5/4}$, it does not exceed it by much. As we mentioned earlier, it might therefore be prone to signing failures and hence might not currently be suitable for SQISign. The next 255-bit prime of mention, $p = 2r^3 - 1$ with $r = 26606682403634464748953600$, is very similar to the previous prime, however the cofactor T exceeds $p^{5/4}$ by a larger margin, so would be less prone to these failures. In this case we have

$$\begin{aligned}
p+1 &= 2^{40} \cdot 5^6 \cdot 11^3 \cdot 47^3 \cdot 67^6 \cdot 101^3 \cdot 113^3 \cdot 137^3 \cdot 277^3 \cdot 307^3 \cdot 421^3, \text{ and} \\
p-1 &= 2 \cdot 3^2 \cdot 19^3 \cdot 37 \cdot 59 \cdot 61 \cdot 97 \cdot 181^2 \cdot 197 \cdot 223 \cdot 271 \cdot 281 \cdot 311 \cdot 397 \cdot 547 \\
&\quad \cdot 1015234718965008560203 \cdot 3143438922304814418457.
\end{aligned}$$

We additionally ran experiments with the 64-bit twins produced from CHM with $B = 547$ and found the following 253-bit prime $p = 2r^4 - 1$ with $r = 8077251317941145600$. Among all the primes that we found for NIST-I security, this appears to be the best. It has both a larger power of two compared to the primes mentioned above found with $n = 3$ and a smaller smoothness bound – thus making the signing cost metric attractively small. Additionally the cofactor T is large enough to be practical for SQISign without any failures. We note once again that this prime would have been out of scope for the authors of [DFLLW23] to find since they constrained their search to only find primes for which the power of two is larger than the one found here. We have

$$\begin{aligned}
p+1 &= 2^{49} \cdot 5^8 \cdot 13^4 \cdot 41^4 \cdot 71^4 \cdot 113^4 \cdot 181^4 \cdot 223^4 \cdot 457^4, \text{ and} \\
p-1 &= 2 \cdot 3^2 \cdot 7^5 \cdot 17 \cdot 31 \cdot 53 \cdot 61 \cdot 73 \cdot 83 \cdot 127 \cdot 149 \cdot 233 \cdot 293 \cdot 313 \cdot 347 \cdot 397 \\
&\quad \cdot 467 \cdot 479 \cdot 991 \cdot 1667 \cdot 19813 \cdot 211229 \cdot 107155419089 \\
&\quad \cdot 295288804621
\end{aligned}$$

NIST-III parameters. We targeted 384-bit primes using $n = 3, 4$ and 6. The challenge in all three of these scenarios is finding enough twins whose product is divisible by a large power of two. With the limited yield of 128-bit twins, finding such primes is not straightforward; the example with $n = 3$ in Table 4 is the only such instance that we managed to find. The picture is somewhat similar with the 96-bit twins: while we have more of them, the success probabilities in Table 3 suggest that we need a lot more twins with a large power of two in order to produce any SQISign-friendly instances. One exceptional prime that was found in this search was the following 375-bit prime, $p = 2r^4 - 1$ with $r = 12326212283367463507272925184$. Here we have

$$\begin{aligned}
p+1 &= 2^{77} \cdot 11^4 \cdot 29^4 \cdot 59^4 \cdot 67^4 \cdot 149^4 \cdot 331^4 \cdot 443^4 \cdot 593^4 \cdot 1091^4 \cdot 1319^4, \text{ and} \\
p-1 &= 2 \cdot 3 \cdot 5 \cdot 13 \cdot 17 \cdot 31 \cdot 37 \cdot 53 \cdot 83 \cdot 109 \cdot 131 \cdot 241 \cdot 269 \cdot 277 \cdot 283 \cdot 353 \cdot 419 \\
&\quad \cdot 499 \cdot 661 \cdot 877 \cdot 1877 \cdot 3709 \cdot 9613 \cdot 44017 \cdot 55967 \cdot 522673 \cdot 3881351 \\
&\quad \cdot 4772069 \cdot 13468517 \cdot 689025829 \cdot 30011417945673766253.
\end{aligned}$$

Of the NIST Level III primes listed in Table 4, the prime that shows the most promise is the 382-bit prime $p = 2r^6 - 1$ with $r = 11896643388662145024$. Not only is the power of two particularly large but also the smoothness bound of the cofactor T is quite small, reflected in its small signing cost metric (when compared to other p where $p^2 - 1$ is divisible by a large power of 2). We have the factorisations

$$\begin{aligned}
p+1 &= 2^{79} \cdot 3^6 \cdot 23^{12} \cdot 107^6 \cdot 127^6 \cdot 307^6 \cdot 401^6 \cdot 547^6, \text{ and} \\
p-1 &= 2 \cdot 5^2 \cdot 7 \cdot 11 \cdot 17 \cdot 19 \cdot 47 \cdot 71 \cdot 79 \cdot 109 \cdot 149 \cdot 229 \cdot 269 \cdot 283 \cdot 349 \cdot 449 \\
&\quad \cdot 463 \cdot 1019 \cdot 1033 \cdot 1657 \cdot 2179 \cdot 2293 \cdot 4099 \cdot 5119 \cdot 10243 \cdot 381343 \\
&\quad \cdot 19115518067 \cdot 7408818089724412333 \cdot 8323214379148213516392 1.
\end{aligned}$$

NIST-V parameters. We targeted 512-bit primes using $n = 4$ and 6. Once again, combining our CHM runs with $n = 6$ proved to be the best option for finding SQISign parameters at this level. None of the twins found at the 128-bit level combined with $n = 4$ to produce any SQISign friendly primes. From the set of 85-bit twins found in the $B = 547$ CHM run, the 510-bit prime $p = 2r^6 - 1$ with $r = 319297404279448700065218 56$ is particularly attractive. The power of two here is the largest found from this run. Here we have

$$\begin{aligned}
p+1 &= 2^{91} \cdot 19^6 \cdot 61^6 \cdot 89^6 \cdot 101^6 \cdot 139^6 \cdot 179^6 \cdot 223^6 \cdot 239^6 \cdot 251^6 \cdot 281^6, \text{ and} \\
p-1 &= 2 \cdot 3^2 \cdot 5 \cdot 7 \cdot 13 \cdot 23 \cdot 29 \cdot 31 \cdot 41 \cdot 53 \cdot 109 \cdot 149 \cdot 157 \cdot 181 \cdot 269 \cdot 317 \cdot 331 \\
&\quad \cdot 463 \cdot 557 \cdot 727 \cdot 10639 \cdot 31123 \cdot 78583 \cdot 399739 \cdot 545371 \cdot 550657 \cdot 4291141 \\
&\quad \cdot 32208313 \cdot 47148917 \cdot 69050951 \cdot 39618707467 \cdot 220678058317 \\
&\quad \cdot 107810984992771213 \cdot 177993780932160825 7.
\end{aligned}$$

The 85-bit twins found in the CHM run with $B = 2^{11}$ were used to try and find NIST-V parameters. The largest power of two that was found in this run which is suitable for SQISign was $f = 109$. The prime with smallest signing cost metric while having a relatively large power of two is the following 508-bit prime, $p = 2r^6 - 1$ where $r = 26697973900446483680608256$. Here we have

$$\begin{aligned}
p+1 &= 2^{85} \cdot 17^{12} \cdot 37^6 \cdot 59^6 \cdot 97^6 \cdot 233^6 \cdot 311^{12} \cdot 911^6 \cdot 1297^6, \text{ and} \\
p-1 &= 2 \cdot 3^2 \cdot 5 \cdot 7 \cdot 11^2 \cdot 23^2 \cdot 29 \cdot 127 \cdot 163 \cdot 173 \cdot 191 \cdot 193 \cdot 211 \cdot 277 \cdot 347 \cdot 617 \\
&\quad \cdot 661 \cdot 761 \cdot 1039 \cdot 4637 \cdot 5821 \cdot 15649 \cdot 19139 \cdot 143443 \cdot 150151 \cdot 3813769 \\
&\quad \cdot 358244059 \cdot 992456937347 \cdot 35324048178196536982389750 7 \\
&\quad \cdot 86010200695145744013716588914030 21.
\end{aligned}$$

Comparison to prior work. The state-of-the-art implementation of SQISign uses a 254-bit prime that was found using the extended Euclidean algorithm (XGCD) [Cos20, DFKL+20] (see Section VI.2). With this method, it is possible to, for example, force $p \pm 1$ and $p \mp 1$ to be divisible by a large power of 2 and 3 (respectively). Indeed, with this approach, a smooth factor of size

NIST security level		p	$\lceil \log_2(p) \rceil$	f	B	$\sqrt{B}/f$	$\log_p(T)$
		p_{3923} [DFLLW23]	254	65	3923	0.96	1.32
	n	r					
NIST-I	2	1211460311716772790566574529001291776	241	49	1091	0.67	1.28
		2091023014142971802357816084152713216	243	49	887	0.61	1.28
	3	3474272816789867297357824	246	43	547	0.54	1.29
		10227318375788227199589376	251	31	383	0.63	1.31
		21611736033260878876800000	254	31	421	0.66	1.28
		20461449125500374748856320	254	46	523	0.50	1.26
		26606682403634464748953600	255	40	547	0.58	1.28
	4	1466873880764125184	243	49	701	0.54	1.28
		8077251317941145600	253	49	479	0.45	1.30
		12105439990105079808 [DFLLW23]	255	61	1877	0.71	1.31
		13470906659953016832 [DFLLW23]	256	61	1487	0.63	1.30
NIST-III	3	1374002035005713149550405343373848576	362	37	1277	0.97	1.25
	4	5139734876262390964070873088	370	45	11789	2.41	1.26
		12326212283367463507272925184	375	77	55967	3.07	1.31
		18080754980295452456023326720	377	61	95569	5.07	1.26
		27464400309146790228660255744	379	41	13127	2.79	1.29
	6	2628583629218279424	369	73	13219	1.58	1.27
		5417690118774595584	375	79	58153	3.05	1.27
		11896643388662145024	382	79	10243	1.28	1.30
	12	5114946480 [DFDdSGF+21]	389	49	31327	3.61	1.30
NIST-V	6	9469787780580604464332800	499	109	703981	7.70	1.25
		12233468605740686007808000	502	73	376963	8.41	1.28
		26697973900446483680608256	508	85	150151	4.56	1.26
		31929740427944870006521856	510	91	550657	8.15	1.25
		41340248200900819056793600	512	67	224911	7.08	1.28

Table 4: A table of SQISign parameters $p = p_n(r)$ for twin-smooth integers $(r, r \pm 1)$ found using CHM at each security level. The f is the power of two dividing $(p^2 - 1)/2$ and B is the smoothness bound of the odd cofactor $T \approx p^{5/4}$. It also includes existing primes in the literature including the state-of-the-art.

$\approx \sqrt{p}$ comes for free in both $p \pm 1$. Concretely, the prime p used in [DFLLW23] has

$$\begin{aligned} p + 1 &= 2^{65} \cdot 5^2 \cdot 7 \cdot 11 \cdot 19 \cdot 29^2 \cdot 37^2 \cdot 47 \cdot 197 \cdot 263 \cdot 281 \cdot 461 \cdot 521 \cdot 3923 \cdot 62731 \\ &\quad \cdot 96362257 \cdot 3924006112952623, \text{ and} \\ p - 1 &= 2 \cdot 3^{65} \cdot 13 \cdot 17 \cdot 43 \cdot 79 \cdot 157 \cdot 239 \cdot 271 \cdot 283 \cdot 307 \cdot 563 \cdot 599 \cdot 607 \cdot 619 \\ &\quad \cdot 743 \cdot 827 \cdot 941 \cdot 2357 \cdot 10069. \end{aligned}$$

This prime features a larger power of two compared to the primes that are mentioned in Table 4 and target NIST-I security. On the other hand, the cofactor T has a larger smoothness bound, $B = 3923$, and a larger signing cost metric, at 0.96. Given that these two latter quantities are larger than in the primes we give, one could be tempted to think that the signing algorithm would perform better when instantiated with our primes. However, the implementation in [DFLLW23], using the state-of-the-art prime above, benefits from the large power of three and the fact that most of the smooth factors are relatively small. It is therefore unclear whether our NIST-I primes would outperform the prime above. Evaluating this in detail would require an implementation comparison, see Remark VI.22.

As mentioned earlier, our work showcases the first credible primes for SQISign at the NIST-III and NIST-V security level. A beneficial feature about most of the primes found in Table 4 is that the majority of the smooth factors are relatively small (e.g. $B < 2^{10}$). In comparison, we expect the XGCD method to scale worse for larger security levels, i.e., requiring much larger smoothness bounds. This is similar to the analysis in [CMN21], which shows that while the XGCD approach has reasonable smoothness probabilities for NIST-I parameters, other methods become superior for larger sizes.

We note that there are other 384 and 512-bit primes in the literature for which $p^2 - 1$ is smooth [CMN21, DFDdSGF$^+$21]. None of the primes from [CMN21] have a large enough power of two for a suitable SQISign application. Some primes were found in the context of the isogeny-based public-key encryption scheme Séta [DFDdSGF$^+$21] that could be suitable for SQISign. As part of their parameter setup, they required finding $\approx$ 384-bit primes[6]. Of the 7 primes that they found, the 389-bit prime, $p = 2r^{12} - 1$ with $r = 5114946480$ appears to be somewhat SQISign-friendly to achieve NIST-III security (see Table 4). However, in addition to its worse signing metric, representations of $\mathbb{F}_p$-values require an additional register in this case compared to our primes of bitlengths slightly below 384. Thus, we can expect implementations of $\mathbb{F}_p$-arithmetic to perform significantly worse for this prime.

Remark VI.13. *The requirement we impose on $p^2 - 1$ being divisible by $2^f \cdot T$ is to ensure that it fits within the current implementation of SQISign. At present, the SQISign implementation has a fine-grained optimisation of their ideal to isogeny algorithm to the setting with $\ell = 2$. In general, one could instead allow $p^2 - 1$ to be divisible by $L \cdot T$, for a, smooth number L with* $\gcd(L, T) = 1$. *This could open new avenues to find SQISign-friendly primes, but would require a reconfiguration of the SQISign code. For example, using the prime $p = p_2(r)$ found with $r =$* 20910230141429718023578160841527132l6 *from Table 4, we could use $L = 2^{49} \cdot 3^4 \cdot 5 \mid p^2 - 1$ and still have a large enough smooth factor T to exceed $p^{5/4}$.*

VI.6 Smooth Twins using the Extended Euclidean Algorithm over Polynomial Rings

As mentioned at the beginning of this Chapter, these last two section represent new work that is not presented in [BSC$^+$22].

In Sections VI.2 and VI.3, we outlined other known methods for generating smooth twins. Recall that these methods came in two flavours: the first are constructive methods and the other are probabilistic methods. Among the probabilistic methods mentioned in SectionVI.2, the algorithms can be further subdivided into two flavours. There are algorithms that work over the integers including the extended Euclidean algorithm and the Chinese remainder theorem and the other algorithms work over a polynomial ring including searching with $r = x^n - 1$ and searching with ideal PTE solutions. There is a natural way to combine these techniques together using a generalisation of the extended Euclidean algorithm over the polynomial ring $\mathbb{Q}[x]$. In this section we describe this new method and showcase the potential benefits over the prior probabilistic methods as well as demonstrate that there are certain input polynomials that can be used to find twins but cannot be used for cryptographic purposes.

[6]That satisfy some mild conditions outside of just requiring $p^2 - 1$ to be smooth

VI.6.1 General Method

Choose two coprime polynomials, $F, G \in \mathbb{Z}[x]$, that split completely into linear factors and the number of distinct roots of FG is small. These points are not strictly necessary but it would help the smoothness probabilities if they were included. Let A be the set of all roots of FG. Use the extended Euclidean (XGCD) algorithm over the polynomial ring with rational coefficients (see [Sho09, Section 17.4] for a full description of the algorithm) to find the corresponding polynomials $S, T \in \mathbb{Q}[x]$ such that $FS + GT \equiv 1$. Note here that these polynomials will, in the majority of cases, not have integer coefficients. For now we start by assuming that these have integer coefficients and later will discuss way to get around this when they have rational coefficients. Much like in the setting of the integers, one can show that, if at least one of F, G has degree strictly greater than 1, then $\deg(S) < \deg(G)$ and $\deg(T) < \deg(F)$ [Sho09, Theorem 17.4]. Then the polynomials $f := F \cdot S$ and $g := -G \cdot T$ have integer coefficients and differ by 1. Without loss of generality, we can assume that the leading coefficient of these polynomials is positive. Now we sieve through an interval of integers and identify the integers r such that $r - a$ is smooth for all $a \in A$. Then the pair $(f(r), g(r))$ generate twin-smooth integers if and only if $S(r)$ and $T(r)$ are smooth.

Now we deal with the general case with S, T have rational coefficients. To do this we adopt a similar approach as done in the PTE work. Let v be the lowest common multiple of all the denominators of the coefficients of S and T. Then the polynomials vS and vT have integer coefficients and $f := vFS$ and $g := -vGT$ are polynomials that differ by v. Much like before we sieve through an interval of integers and identify the integers r such that not only $r - a$ is smooth for all $a \in A$ but also $f(r) \equiv g(r) \equiv 0 \mod v$. When we are in this setting, then the pair $(f(r)/v, g(r)/v)$ generate twin smooth integers if and only if $vS(r)$ and $vT(r)$ are either smooth integers or the only rough factors that may exist only divide v. It is worth noting that in the vast majority of examples that we are interested in v will either be small or be already smooth.

It is worth noting that the resulting polynomials S, T from the extended Euclidean algorithm are the unique polynomials that satisfy $FS+GT \equiv 1$ and $\deg(S) < \deg(G)$ and $\deg(T) < \deg(F)$. Having said this there are other polynomials S', T' of larger degrees $\deg(S') \geq \deg(G)$ and $\deg(T') \geq \deg(F)$ that also satisfy $FS' + GT' \equiv 1$. One can describe the general solution of this equation using the polynomials S and T: namely $S' = S + P \cdot G$ and $T' = T - P \cdot F$ for any polynomial P. We give a name for this procedure of using the general solution and call it perturbing the solution S, T. One can replace S and T with these perturbed solutions S' and T' and attempt to find twin smooth integers. This will inevitable increase the degree of the polynomials $F \cdot S'$ and $-G \cdot T'$ and hence decrease the smoothness probability. This is also apparent in the integer setting but the difference here is that we have flexibility on the smooth input r that we choose. Having said this, for some choice of polynomials F, G, this perturbation step is a necessity rather than an optional add-on. This will be illustrated later in Section VI.6.2.

Remark VI.14. *There are two natural choices for how you choose the polynomials F and G that feed as inputs to the algorithm. The first way is to simply choose random integers a that feed as roots of F and G. The second approach is have a precomputed list of polynomials F and G and choose one of them. This latter approach is similar to that adopted by the authors of the PTE article* *and can offer better smoothness probabilities as it shall be noted shortly. Additionally in the latter approach, we can assume that F and G are chosen such that S and T have not rational roots but can be factored into irreducible factors of degree at least* 2*.*

Realising the generalisation. This naturally generalises the XGCD method over the integers that was mentioned in Section V.2.1. Here we also show how to the polynomial methods for generating twin-smooth integers mentioned in Section V.2.1 fall as a special case to the method just presented.

For any integer n, let $F(x) = x^n$ and $G(x) = x - 1$. The result of the extended Euclidean algorithm outputs the polynomials $S(x) = 1$ and $T(x) = -x^{n-1} - \cdots - x - 1$. Hence one obtains $f(x) = F(x)S(x) = x^n$ and $g(x) = -G(x)T(x) = x^n - 1$ which are the polynomials used in [Cos20] to find such twins. Note that when n is even, we can still obtain these polynomials when choosing $F(x) = x^n$ and $G(x) = x^2 - 1$.

Suppose that $\mathscr{A} = \{a_1, \cdots, a_n\}$ and $\mathscr{B} = \{b_1, \cdots, b_n\}$ is an ideal PTE solution and let C be the constant difference between the two associated polynomials in accordance to Proposition VI.4. Let $\mathscr{A}' = \{a_{i_j}\}$ and $\mathscr{B}' = \{b_{i_j}\}$ be subsets of $\mathscr{A}$ and $\mathscr{B}$ respectively such that $\#\mathscr{A}' + \#\mathscr{B}' > n$. Let $F(x) = \prod_{a \in \mathscr{A}'}(x - a)$ and $G(x) = \prod_{b \in \mathscr{B}'}(x - b)$. Then the result of the extended Euclidean algorithm will output the polynomials $S(x) = \left(\prod_{a \in \mathscr{A} \setminus \mathscr{A}'}(x - a)\right)/C$ and $T(x) = -\left(\prod_{b \in \mathscr{B} \setminus \mathscr{B}'}(x - b)\right)/C$. This is because the polynomials $\prod_{i=1}^{n}(x - a_i)$ and $\prod_{j=1}^{n}(x - b_j)$ differ by C and the result of the extended Euclidean algorithm is unique when we want $\deg(S) < \deg(G)$ and $\deg(T) < \deg(F)$. The requirement on $\#\mathscr{A}' + \#\mathscr{B}' > n$ is so that, without any perturbation, we do get the intended polynomials.

Smoothness probabilities. Recall Heuristic III.7 that says computing the smoothness probability of an evaluated polynomial depends on the irreducible factors of the polynomial. Thus the probability of obtaining twin smooth integers from this method depends on the irreducible factors of F, G, S and T. Given this, one might suggest that the smoothness probabilities would be optimised when the degrees of each of the factors are all 1, in other words the setting when we have an ideal PTE solution. This would certainly be the case when there are repeated factors in either polynomials since there would be fewer smoothness checks to be done. However, such solutions of this type only exist when the degree of the polynomial is $n \in \{2, 3, 4, 6\}$. For other degrees, only distinct linear factors are known in the literature. This suggests the following: Instead of having all of the factors being linear, you replace some of the linear factors with quadratic (or potentially higher degree) factors and counterbalance that by having more square (or potentially larger power) factors. It is relatively straightforward search for such polynomials using the extended Euclidean algorithm technique mentioned above. The idea consists of fixing an integer bound $\kappa > 0$ and enumerate over all completely split and coprime polynomials F and G such that each root in their product, i.e. the set A, is at most κ in absolute value. Computing the polynomials S and T for each pair F and G, attempt to factor them and record it if the smoothness probability is better or on par with that of a corresponding PTE solution. Note that as we would like square (or larger power) factors in the resulting polynomials, the number of distinct roots in A is smaller than the more general case – thus making the enumerated search more effective. Through a small scale search using this technique (with a bound $\kappa = 8$ and F,G

containing at most square factors), one can find the following two polynomials that differ by a constant and factors into linear factors up to one quadratic factor.

$$\begin{aligned}
f(x) &= x^8 + 4x^7 - 198x^6 - 608x^5 + 11953x^4 + 24924x^3 - 207756x^2 - 220320x \\
&= (x-8)(x-5)x(x+1)(x+6)(x+9)(x^2+x-102), \text{ and} \\
g(x) &= x^8 + 4x^7 - 198x^6 - 608x^5 + 11953x^4 + 24924x^3 - 207756x^2 - 220320x + 1166400 \\
&= (x-9)^2(x-3)^2(x+4)^2(x+10)^2.
\end{aligned}$$

Calculating the smoothness probability using the Heuristic III.7 and comparing it to the corresponding probability from an ideal PTE solution of size 8 that exist in the literature, one can see that this smoothness probability from the presented polynomials is better. In addition, here are similar polynomials but of degree 12:

$$\begin{aligned}
f(x) &= x^{12} - 9052x^{10} + 30727014x^8 - 48031637476x^6 + 33806587730449x^4 \\
&\qquad - 8577014182099272x^2 + 74489608030488336 \\
&= (x-49)(x-46)(x-31)(x-3)(x+3)(x+31)(x+46)(x+49) \\
&\qquad (x^2-31x-1302)(x^2+31x-1302), \text{ and} \\
g(x) &= x^{12} - 9052x^{10} + 30727014x^8 - 48031637476x^6 + 33806587730449x^4 \\
&\qquad - 8577014182099272x^2 + 701251897192728336 \\
&= (x-53)^2(x-39)^2(x-14)^2(x+14)^2(x+39)^2(x+53)^2.
\end{aligned}$$

However, as also highlighted in the PTE paper [CMN21], we can loosen the criterion on requiring each side of the twin to be fully smooth. As mentioned in Chapter IV, the cryptographic applications that require this set up do not require everything to be smooth but at the very least require some large enough smooth cofactor. This notably includes SQISign which has been a focus of the chapter. While you would still need to deal with the probabilities that arise from the higher degree polynomials, this could be counterbalanced by requiring a lot fewer linear terms to be smooth which may be repeated. This is similar to the examples mentioned above but one can find better polynomials that can give more overall smoothness from a small input. This will be explored further in Section VI.7 where, with a particular choice of F and G, we use this new technique to find SQISign friendly parameters.

VI.6.2 Choosing incorrect polynomials for cryptographic applications

We have to be careful in how we choose F, G if we wish to use apply these for cryptographic purposes. Here we describe a certain class of polynomials for which, when you apply the extended Euclidean algorithm and sum the corresponding twins, it will never be a prime.

Consider the polynomials $F(x) = (x+1)^n$ and $G(x) = x^n$ for a strictly positive integer n. This might look like an advantageous choice of polynomials for cryptographic purposes since we could guarantee a lot of smoothness from a small amount of smooth input (namely just assuming r and $r+1$ are smooth). However, this choice of polynomial will unfortunately not lead to any cryptographic applications. The reason is that the associated polynomial can be reduced to smaller factors and hence the associated integer resulting from summing the twin-smooth integers, namely $|(r+1)^n S(r) - r^n T(r)|$, will never be a prime. In particular, it will contain $2r+1$ as a factor. This result is not only formalised but also generalised with aid of Proposition VI.15 and applies to a broader range of polynomials. Having said this, perturbing the solutions by some polynomial can give you irreducible polynomials. But this is at a cost of having a larger (and most likely irreducible) factor which decreases the smoothness probabilities. This is the case when the perturbation polynomial is $P(x)=1$.

Proposition VI.15. *For a strictly positive integer n and two polynomials F, G that differ by a non-zero rational, let S_n and T_n be the outputted rational polynomials from applying the extended euclidean algorithm to the polynomials F^n and G^n. Note here that $F^n := F \cdot F \cdots F$ denotes the product of F n times rather than composition. Then there is a constant $C_{n,F,G}$ such that*

$$\begin{aligned} F(x)S_{n+1}(x) - S_n(x) &= C_{n,F,G}G(x)^n(F(x)+G(x)), \quad \textit{and} \\ G(x)T_{n+1}(x) - T_n(x) &= -C_{n,F,G}F(x)^n(F(x)+G(x)). \end{aligned}$$

Moreover, in the setting when $F(x) = x+1$ and $G(x) = x$, the polynomial $H_n(x) := F^n(x)S_n(x) - G^n(x)T_n(x)$ will always have $2x+1$ as a factor.

Proof. Let $m := F - G$ be the rational difference between these polynomials. Since S_n and T_n are polynomials that resulted from applying the extended euclidean algorithm to F^n and G^n we have

$$F(x)^n S_n(x) + G(x)^n T_n(x) = 1, \quad \deg(S_n) = \deg(T_n) < n. \tag{$\ast$}$$

In order to prove the proposition, we need two key ingredients. These are formulated in the following claims.

Claim 1. *For every n, we have the following* $\deg(S_n) = \deg(T_n) = \deg(g)\cdot(n-1)$.

Proof. Let's start by applying the Euclidean algorithm step by step to $F(x)^n$ and $G(x)^n$. Write $F(x)^n = (G(x)+m)^n = G(x)^n + R_1(x)$. It is straight forward to see that $\deg(r_1) = \deg(G)(n-1)$. Now write $G(x)^n = Q_2(x)R_1(x) + R_2(x)$ with $\deg(R_2) < \deg(R_1)$. Again, it is straight forward to see that $\deg(Q_2) = \deg(G)$ and $\deg(R_2) = \deg(G)(n-2)$ since the $x^{\deg(G)(n-2)}$ coefficient of $Q_2(x)R_1(x)$ is non-zero. One can repeat this iteratively and deduce that $\deg(Q_k) = \deg(G)$ and $\deg(R_k) = \deg(G)(n-k)$ for all $2 \le k \le n$.

Using these polynomials, we can apply the extended Euclidean algorithm to recover the polynomials S_n and T_n. In fact, using Theorem 17.4(iv) from [Sho09], we obtain the desired result. □

Claim 2. *For every n, we have the following closed forms for S_n and T_n:*

$$\begin{aligned} S_n(x) &= m^{-n}\sum_{k=0}^{n-1}\binom{n+k-1}{k}(-G(x)/m)^k, \textit{ and} \\ T_n(x) &= (-m)^{-n}\sum_{k=0}^{n-1}\binom{n+k-1}{k}(F(x)/m)^k. \end{aligned}$$

Proof. Let $p_n(x) = m^{-n}\sum_{k=0}^{n-1}\binom{n+k-1}{k}(-G(x)/m)^k$. Since $\deg(S_n) = \deg(p_n) = \deg(g)(n-1)$ then, by the uniqueness of S_n, T_n, it suffices to show that $F(x)^n p_n(x) - 1$ is divisible by $G(x)^n$.

Firstly, write $F(x) = m(G(x)/m+1)$. When multiplying $F(x)^n$ and $p_n(x)$, write the result as a polynomial in $G(x)$ - so we have $F(x)^n p_n(x) = \sum_{k=0}^{2n-1} a_k G(x)^k$. For $k < n$ the coefficient a_k in this product is equal to

$$m^{-k} \sum_{i=0}^{k} \binom{n}{i} \binom{n+k-i-1}{k-i} (-1)^{k-i}.$$

When $k = 0$ this is equal to 1. As a simple exercise in using the "Upper Negation" and Vandermonde's identities for binomial coefficients, when $0 < k < n$ this is equal to 0. This proves the closed form of S_n and proving the closed form of T_n can be done with a similar strategy. □

With the closed form for S_n obtained in Claim 2, it is an algebraic exercise to obtain the first of these formulas. In the process one shows that $C_{n,F,G} = m^{-2n-1}(-1)^n \binom{2n-1}{n-1}$. One could do the same algebraic exercise for T_n but by considering the equation $(*)$ for n and $n+1$, we have

$$F(x)^n \left(F(x) S_{n+1}(x) - S_n(x)\right) = G(x)^n \left(T_n - G(x) T_{n+1}(x)\right).$$

This shows that the second of these formulas can easily be found from the first.

Now suppose that $F(x) = x+1$ and $G(x) = x$. Using the derived recursive formulas for S_n, T_n and by induction, we have $S_n(-1/2) = 2^{n-1}$ and $T_n(-1/2) = -(-2)^{n-1}$. Plugging $-1/2$ into the defining formula for H_n gives the result.

□

Remark VI.16. *The concluding statement in the proposition is not specific to $F(x) = x+1$ and $G(x) = x$ and it applies more generally. In the general setting the factor that appears is the polynomial $F+G$. As another remark, it is straight forward to adopt the arguments addressed above to the setting when F is a linear transform of G (i.e. $F(x) = aG(x)+b$)*

This factoring property that one might get from these polynomials appears to be a somewhat global property. Let's describe what we mean by this. For any two polynomials F, G, with the only restriction on them this time is that $\gcd(F,G) = 1$, let

$$\begin{aligned} \tilde{H_n}(x) &:= F(x)^n(S_n(x) + P(x)G(x)^n) - G(x)^n(T_n(x) - P(x)F(x)^n) \\ &= 2F(x)^n(S_n(x) + P(x)G(x)^n) - 1 \\ &= 2G(x)^n(T_n(x) - P(x)F(x)^n) + 1 \end{aligned}$$

where S_n, T_n are once again the result of applying the extended Euclidean algorithm to F^n, G^n and we perturb the solution by a polynomial P. Then it appears that either all of H_n are reducible (with the possible exception of $\tilde{H_1}$) or eventually all $\tilde{H_n}$ are irreducible. The above proposition captures this in former setting and there are countless examples that captures the latter. For instance, choosing $F(x) = x^3 + x^2 + x + 2$ and $G(x) = x$, then $\tilde{H_1}, \tilde{H_2}, \tilde{H_4}$ are reducible and all subsequence polynomials appear to be irreducible. We formalise this and state it in the following conjecture.

Conjecture VI.17. *Let $F, G, P, S_n, T_n, \tilde{H_n}$ be the polynomials as described above. Suppose there is some $k > 1$ such that $\tilde{H_k}$ is irreducible and assume that this k is minimal. Then there is some $N \geq k$ such that, for all $n \geq N$, $\tilde{H_n}$ is irreducible over the rationals.*

We briefly note that there are other families of rational polynomial that are conjectured to satisfy an irreducible property similar to that of these polynomials $\tilde{H_n}$ [FFK00].

VI.6.3 New technique for finding ideal PTE solutions

Earlier we saw that the method of finding twin smooth integers using PTE solutions appears as a special case of the idea using the extended Euclidean algorithm. If we reverse this thinking, it would seem to suggest that we could use the extended Euclidean algorithm over the polynomial ring to generate ideal PTE solutions.

The idea is the following: take two polynomials, F, G, that completely split over the integers and apply the extended Euclidean algorithm to these polynomials to get the two new polynomials S, T. If $S \cdot T$ completely splits over the integers (up to a rational scaling factor) then with aid of Proposition VI.4 one gets an ideal PTE solution of the form $\text{Roots}(F \cdot S) =_{n-1} \text{Roots}(G \cdot T)$ with $n = \deg(F \cdot S) = \deg(G \cdot T)$. It may be the case that $S \cdot T$ completely splits over the rationals but may not have integer roots. In this case one can simply apply a linear transformation to each of the polynomials to turn it into one with integer roots (see [CMN21, Corollary 1]). While there is an extensive collection of PTE solutions out there, this idea has been utilised to find new solutions.

New parametrised ideal PTE solutions. Here we present a new parametrised family of size 4 ideal PTE solutions that feature one repetition in both sets of integers. Such PTE solutions were considered to find smooth B-SIDH parameters before the polynomial time attacks on SIDH surfaced (mentioned in Section IV.4.2). There one requires to find primes p such that there are large smooth cofactors of $p+1$ and $p-1$ which are roughly the same size. The repetition in both sides of these PTE solutions would have made our chances of finding such parameters more profitable.

While there are complete parametric ideal solutions for a few sizes including 4, understanding when we get solutions with repetition from these formulas is a little non-trivial. Moreover, among the various database sources [Shu01, Cal13, Res21], there are no reported findings of such ideal solutions. We will use the idea mentioned above as a tool to obtain this parametrisation. We firstly note down how this parametrisation looks like.

Proposition VI.18. *Let α, β be rational parameters and*

$$\begin{aligned} a &= \alpha\beta(\beta+1)(\beta^2-2\beta+3), \\ b &= \alpha\beta(\beta+1)(\beta^2+1), \\ c &= \alpha(\beta^2+1)(\beta^2+2\beta-1), \\ d &= \alpha(\beta+1)(\beta-1)^3, \\ e &= 4\alpha\beta \\ C &= \alpha^4\beta^3(\beta-1)^3(\beta+1)^3(\beta^2+1)^2. \end{aligned}$$

Then we obtain an ideal PTE solution of size 4 of the following form: $[0, a, a, c] =_3 [b, b, d, e]$. The quantity C is the constant difference between the corresponding polynomials in accordance to Proposition VI.4.

For concreteness of an example, when choosing $\alpha = -16$ and $\beta = -1/2$, one gets $a = 17$, $b = 5$, $c = 35$, $d = 27$ and $e = 32$. One can easily show that, among all such PTE solutions for which a, b, c, d, e are all positive integers and all coprime, this is the smallest solution – as emphasised by the solutions listed in Table 5.

Proof. We start out by applying the XGCD algorithm to the polynomials $F(x) := x(x+a)^2$ and $G(x) := (x+b)^2$ where a, b are treated as distinct non-zero parameters. This results in the polynomials S, T such that $\deg(S) = 1$ and $\deg(T) = 2$ which are

$$\begin{aligned} S(x) &= \frac{-a+3b}{b^2(a-b)^3}x + \frac{-2a+4b}{b(a-b)^3} \\ T(x) &= \frac{a-3b}{b^2(a-b)^3}x^2 + \frac{2a^2-6ab+2b^2}{b^2(a-b)^3}x + \frac{1}{b^2}. \end{aligned}$$

Since the polynomial s is linear it only suffices to check when t factors. This happens only when its discriminant is a square: $\operatorname{disc}(T) = D^2$. If we let $\mathbf{a} = b(a-b)^3D$, $\mathbf{b} = 2a-4b$ and $\mathbf{c} = 2b$, then this is equivalent to solving the equation

$$\mathbf{a}^2 + \mathbf{b}^2 = 2\mathbf{c}^2.$$

The following claim derives a parametrisation of this equation over any field k but, for our specific purpose, we will restrict to $k = \mathbb{Q}$.

Claim 3. *Over a field k, the general solution to the equation* $\mathbf{a}^2+\mathbf{b}^2 = 2\mathbf{c}^2$ *can be parametrised by two variables* $\alpha, \beta \in k$ *as* $\mathbf{a} = \alpha(\beta^2-2\beta-1)$, $\mathbf{b} = -\alpha(\beta^2+2\beta-1)$ *and* $\mathbf{c} = \alpha(\beta^2+1)$.

Proof. The equation we intend to solve can be considered as an equation over the projective space $\mathbb{P}^2 = \mathbb{P}^2(k)$. This equation has a trivial solution in $[1:1:1] \in \mathbb{P}^2$. If $[u:v:1] \in \mathbb{P}^2$ is a solution that is different from $[1:1:1]$, then the line that joins $[u:v:1]$ and $[1:1:1]$ has a slope $\beta \in k$. Conversely, given $\beta \in k$, there is a line that passes through the point $[1:1:1]$ and has slope β. More precisely, this line has a projective equation of the form

$$\mathbf{b} = \beta(\mathbf{a}-\mathbf{c}) + \mathbf{c}.$$

This line intersects intersects the conic at two points: the first point is $[1:1:1]$ and the second point is $[u:v:1]$ where

$$\begin{aligned} u &= \frac{\beta^2-2\beta-1}{\beta^2+1}, \text{ and} \\ v &= \frac{-\beta^2-2\beta+1}{\beta^2+1}. \end{aligned}$$

Equivalently, this point is $[u:v:1] = [\beta^2-2\beta-1 : -\beta^2-2\beta+1 : \beta^2+1]$. Thus we obtain the parametrisation $\mathbf{a} = \alpha(\beta^2-2\beta-1)$, $\mathbf{b} = -\alpha(\beta^2+2\beta-1)$ and $\mathbf{c} = \alpha(\beta^2+1)$ for $\alpha, \beta \in k$ to the equation $\mathbf{a}^2 + \mathbf{b}^2 = 2\mathbf{c}^2$. □

We turn back to studying this equation over the rationals. For the purpose simplying the expressions later on, we write the general solution to this equation as $\mathbf{a} = 2\alpha(\beta^2-2\beta-1)$, $\mathbf{b} = -2\alpha(\beta^2+2\beta-1)$ and $\mathbf{c} = 2\alpha(\beta^2+1)$ which can be done since α is simply an arbitrary scalar. From this recovering what a, b are in this context is straightforward which are $a = \alpha(\beta^2-2\beta+3)$ and $b = \alpha(\beta^2+1)$. These expressions for a, b are not quite what is stated in the proposition. The

a	b	c	d	e
17	5	35	27	32
86	26	221	125	216
171	51	391	256	375
243	75	775	343	768
524	164	2009	729	2000
594	174	1189	1000	1029
605	185	1739	864	1715
965	305	4331	1331	4320
1463	455	5135	2048	5103
1602	510	8245	2197	8232
1790	530	3869	2744	3645
2471	791	14351	3375	14336
2628	780	5785	3993	5488
2889	909	12019	4000	11979
3608	1160	23345	4913	23328
3735	1095	7519	6144	6655
3962	1190	9605	5832	9317
4455	1335	10591	6591	10240
5027	1595	24215	6912	24167
5049	1629	36019	6859	36000
6620	1940	13289	10985	11664
6830	2210	53261	9261	53240
7398	2250	20125	10648	19773
7749	2289	16459	12000	15379
8021	2561	43931	10976	43875
8987	2915	76055	12167	76032
10269	3129	28459	14739	28000
11556	3756	105481	15625	105456
12015	3855	73759	16384	73695
12386	3806	37541	17576	37125
13076	3836	26441	21296	23625
14472	4440	43105	20577	42592
14573	4745	142715	19683	142688
15930	4710	34069	24565	31944
17153	5525	116675	23328	116603
18074	5894	189029	24389	189000
19214	5954	64349	27000	63869
20195	5915	40391	34391	34560
22095	7215	245791	29791	245760
22473	6765	55555	32928	54043
22572	6660	47545	35152	44217
22715	6755	50759	34295	48384
23579	7619	176039	32000	175959
26010	8070	88501	36501	87880
26672	8720	314465	35937	314432
28170	8790	103429	39304	102885
29358	8610	59245	48013	52728
31160	9320	72929	46305	70304
31437	10185	255595	42592	255507
31841	10421	396611	42875	396576
33561	10461	121411	46875	120736
33885	9945	68731	54880	61731
34047	10335	90895	49152	89167
35684	10604	79289	54000	75449
37638	12330	493885	50653	493848
39542	12410	158045	54872	157437
40871	13271	359471	55296	359375
41445	12465	101659	60835	98784
44099	14459	608039	59319	608000
51260	16820	740921	68921	740880
52025	16925	492179	70304	492075
52415	15455	109871	81920	101871
52767	16575	213775	73167	212992
52988	15860	124745	78608	120393
53618	16910	231845	74088	231173
54824	16184	115889	85169	108000
59157	19425	894475	79507	894432
64638	20370	275965	89373	275128
65043	21195	658615	87808	658503
65583	19215	131455	109503	114688
67779	20859	208999	96000	206839
67826	22286	1070741	91125	1070696
68255	20735	183599	98415	180224
70686	22386	328861	97336	328125
71631	21231	156031	109375	147456
73062	21450	148525	117912	133837
75060	22620	187369	109744	182505
76505	22685	167171	116640	158171
77303	25415	1271855	103823	1271808
80069	26129	864059	108000	863939
85140	25980	239449	121945	235824
86616	25416	175441	140625	157216
87624	28824	1500049	117649	1500000
90801	28101	299251	127776	296875
91034	28934	453509	125000	452709
91490	27230	205781	137781	196520
93456	29616	439921	128625	438976
97247	31775	1114175	131072	1114047
98021	28721	196571	163296	171875
98825	32525	1757651	132651	1757600
102476	31076	271001	148176	265625
104585	32045	307139	148955	303264
105066	30966	219541	164616	203125
110619	35139	544999	151959	544000
110942	36530	2047085	148877	2047032
114653	34265	266555	170723	256608
114950	36650	610589	157464	609725
116721	38181	1415011	157216	1414875
124011	40851	2370871	166375	2370816
126770	38990	388229	179685	384104
127688	37400	255425	216513	219488
135812	41420	379865	194672	373977
138068	45500	2731625	185193	2731568
138635	45395	1772999	186624	1772855
139139	40859	283319	224000	255879
140670	41490	295501	219501	274360
141372	42420	339865	208537	329232
142722	45630	805285	195112	804357
144245	43265	345611	212960	334611
150993	48165	810355	206763	809248
151317	47265	563755	210912	560947
153149	50489	3132059	205379	3132000
162459	47619	326599	268279	288000
163133	53465	2194955	219488	2194803
169290	55830	3574981	226981	3574920
171899	51779	427319	251559	416000
174339	51579	374599	268119	352000
174420	55740	974521	238521	973360
174638	55970	1043165	238328	1042173
175644	53844	518569	250000	512169
179192	52520	360065	296352	317057
180080	55760	597329	253265	592704
186527	61535	4063295	250047	4063232
189335	57095	475391	276480	463391
189675	59475	747799	263424	744775
189945	55965	396019	298144	365835
190359	62439	2688079	256000	2687919
193698	57630	434485	292008	414613

Table 5: A list consisting of all inequivalent and normalised sized 4 *ideal PTE solutions of the form* $[0, a, a, c] =_3 [b, b, d, e]$ *with* $0 < b < a < 200000$ *and* $d < e$.

reason is because to turn it into an ideal PTE solution we need the resulting polynomials, $-(F \cdot S)(x), (G \cdot T)(x)$, to have integral coefficients and be monic in accordance to Proposition VI.4. Currently the leading coefficient of these polynomials is $(a-3b)/(b^2(a-b)^3) = \beta(\beta+1)/(\alpha^4(\beta-1)^3(\beta^2+1)^2)$. To do this we first apply the linear transformation $x \mapsto x/(\beta(\beta+1))$ and then multiply these polynomials through by $C = \alpha^4\beta^3(\beta-1)^3(\beta+1)^3(\beta^2+1)^2$. This makes these polynomials have integral coefficients and be monic. After doing all the algebra, the expressions for a, b, c, d, e materialise as stated in the proposition. □

Remark VI.19. *The strategy laid out in the proof of the Proposition can be generalised in order to obtain a complete parametrisation of all ideal PTE solutions of size* 4 *not just those with this specific shape.*

Corollary VI.20. *Suppose we have an ideal PTE solution of the form* $[0, a, a, c] =_3 [b, b, d, e]$ *with* $a > b$ *and* $a, b, c, d, e > 0$. *Then we have*

$$3b < a < (2+\sqrt{2})b.$$

Proof. By the parametrisation of such solutions given in Proposition VI.18, we have

$$\frac{a}{b} = \frac{\beta^2 - 2\beta + 3}{\beta^2 + 1},$$

for some $\beta \in \mathbb{Q}$. As a rational function, the right hand expression attains a global maximum at $\beta = 1 - \sqrt{2}$. Thus, after evaluation, we get $a/b < 2 + \sqrt{2}$ which proves the upper bound.

For the lower bound, suppose that $b < a \leq 3b$. Once again, substitute the parametric expressions for a and b. After solving the inequality, one deduces that $\alpha \geq 0$ and either $\beta < -1$ or $0 < \beta < 1$.

Recall that d and e can be written in terms of this parametrisation as $d = \alpha(\beta+1)(\beta-1)^3$ and $e = 4\alpha\beta$. If $\beta < -1$ then, since $\alpha \geq 0$, we must have $e = 4\alpha\beta < 0$. Similarly, if $0 < \beta < 1$ then $d = \alpha(\beta+1)(\beta-1)^3 < 0$. In either case, we get a contradiction to the positivity of d and e and thus proves the intended lower bound. □

As a consequence of the above proof, we must have $\alpha < 0$ and $-1 < \beta < 0$. Hence we can write $\beta = -p_0/q_0$ for some positive coprime integers p_0, q_0 with $p_0 < q_0$. Moreover, if the PTE solution is normalised in the sense that not only does it satisfy the condition given in the above Corollary but also a, b, c, d, e are integers such that $\gcd(a, b, c, d, e) = 1$, then we must have $\alpha = -q_0^4$. We note that this is a necessary condition to find such normalised solutions but is not sufficient. Substituting these in, one gets an integral parametrisation of such PTE solutions rather than a rational one.

Using this parameterisation with the help of the bounds given in Corollary VI.20, one can find concrete PTE solutions of this type. Table 5 lists all possible solutions of this type such that $0 < b < a < 200000$.

VI.7 Combining XGCD over polynomial rings with CHM: Theory and Results

Much like what we did with the polynomials $p_n(x) = 2x^n - 1$ in Section Section VI.5, it is a natural question to ask whether we could combine this new technique with the CHM machinery to obtain any meaningful SQISign parameters. Recall this means that you take a smooth twin $(r, r+1)$ and you lift if up through a polynomial in such a way that contains both r and $r+1$ as factors. This lifting is done to a size suitable for cryptographic applications. The great advantage of using the polynomials $p_n(x)$ is that the amount of guaranteed smoothness one can get from a pair of twin smooth integers is quite high. We shall show that for a given degree, this is the optimal amount of guaranteed smoothness one can get. Having said this there are some other polynomials that can give this optimal amount of guaranteed smoothness from just a smooth twin $(r, r+1)$.

Recall that $p_n(x)$ can be obtained from applying the XGCD to the polynomials x^n and $x-1$. The natural next choice is to try the polynomials[7] x^i and $(x+1)^j$ for integers $i, j \geq 2$. As mentioned earlier we constrain ourself to the setting with $i \neq j$ since he have SQISign applications in mind.

Let $S_{i,j}, T_{i,j}$ be the polynomials one gets from applying the extended Euclidean algorithm to the polynomials x^i and $(x+1)^j$. Through a similar method as deployed in the proof of Claim 2, we have the following algebraic expressions for $S_{i,j}$ and $T_{i,j}$

$$S_{i,j}(x) = (-1)^i \sum_{k=0}^{j-1} \binom{i+k-1}{k} (x+1)^k, \quad \text{and}$$
$$T_{i,j}(x) = \sum_{k=0}^{i-1} (-1)^k \binom{j+k-1}{k} x^k.$$

These polynomials have integer coefficients, hence the polynomials $x^i S_{i,j}(x), (x+1)^j T_{i,j}(x)$ differ by one. We set $p_{i,j}(x)$ to be the resulting polynomial with a positive leading coefficient from summing the corresponding twins. In other words, we have

$$\begin{aligned} p_{i,j}(x) &:= (-1)^i \left(x^i S_{i,j}(x) - (x+1)^j T_{i,j}(x) \right) \\ &= (-1)^i \left(2x^i S_{i,j}(x) - 1 \right) \\ &= (-1)^{i+1} \left(2(x+1)^j T_{i,j}(x) + 1 \right). \end{aligned}$$

We will use these polynomials to find primes to attempt to find SQISign friendly parameters through a similar procedure as deployed for the polynomials p_n. Namely, take a smooth twin $(r, r+1)$ and compute the evaluation $p_{i,j}(r)$. If there is enough smoothness from the other factors and it is prime then we get such a parameter. We remark that the amount of guaranteed smoothness from a smooth twin is exactly the same as for the polynomials p_n. Moreover, since $S_{i,j}$ and $T_{i,j}$ do not contain either 0 or -1 as rational roots, then by the uniqueness of the polynomials $S_{i,j}$ and $T_{i,j}$ this is the maximum possible amount of guaranteed smoothness from a single twin. We summarise this result in the following lemma.

Lemma VI.21. *Let $i, j, n \in \mathbb{Z}$ be integers and $P \in \mathbb{Q}[x]$ be a polynomial of degree n and set $Q := P(P+1) \in \mathbb{Q}[x]$. If $x^i(x+1)^j \mid Q(x)$ then $i + j \leq n+1$.*

We first note the special cases when $i = 1$ and $j = 1$ (respectively): $p_{1,n}(x) = p_n(x+1)$ and $p_{n,1}(x) = 2x^n + (-1)^{n+1}$. As alluded earlier, the latter only gives us $p_n(x)$ when n is even. For

[7]The polynomials x^i and $(x-1)^j$ might seem like a more natural choice to generalise the discussion with p_n. However we note that it is actually the same as the choice we make up to applying a linear shift and swapping the exponents i and j.

$i, j \geq 2$, these choice of polynomials seem to be give an interesting alternative to the polynomials p_n. Once again, we limit ourselves to small i, j for the same reason as why we kept n small in our exploration with $p_n(x)$. Additionally, the polynomials $S_{i,j}, T_{i,j}$ appear to be irreducible for all $i, j \geq 2$. Therefore the smoothness probabilities would not be in our favour. In addition, since the maximum exponent of i, j is smaller than its counterpart from $p_n(x)$, then we would require the twin, $(r, r+1)$, to have a much larger power of two for the necessary SQISign requirements. This reduces the space search down quite a bit but doesn't limit the possible scope for finding SQISign parameters. We used our data from Section VI.4.4 and lifted them using new these polynomials to see if the result could be suitable for SQISign. A collection of primes can be found in Table 6. We note that the primes is very comparable to the corresponding primes, $p = p_n(r)$, mentioned in Table 4 – not only with respect to the power of two but also the signing cost metric.

VI.7.1 Results

We start by looking at the case when $i = 3$ and $j = 1$. Here we have $p_{3,1}(x) = 2x^3 + 1$. The factorisation of $p_{3,1}(x) + 1$ is $(x + 1)(x^2 - x + 1)$ which is very similar to the factorisation of $p_3(x) - 1$. This makes the probability of attaining the neccessary amount of smoothness for SQISign given that $r, r + 1$ are smooth the same as mentioned in Table 3. From the collection of 85-bit twins found with the CHM algorithm, we found the following 256-bit prime, $p = 2r^3 + 1$ with $r = 31316053921198399970611200$. Here we have

$$\begin{aligned} p + 1 &= 2 \cdot 7 \cdot 31 \cdot 37 \cdot 79 \cdot 163 \cdot 199 \cdot 233^2 \cdot 271 \cdot 313 \cdot 389^2 \cdot 401 \cdot 409 \cdot 491 \\ &\quad \cdot 8258208689449582657 \cdot 3221159033345167520733643 \\ p - 1 &= 2^{37} \cdot 3^{18} \cdot 5^6 \cdot 53^3 \cdot 59^3 \cdot 149^3 \cdot 151^3 \cdot 211^3 \cdot 239^3 \cdot 317^3 \cdot 373^3 \end{aligned}$$

When combining the 128-bit twins found with the CHM with the polynomial $p_{3,1}(x)$, none of them produce any primes suitable for SQISign friendly parameters for NIST Level III security.

Now we look at the cases with $i = 2$ and $j = 3$ (and vice-versa). Here we have

$$\begin{aligned} p_{2,3}(x) &= 2x^2(3x^2 + 8x + 6) - 1 = 2(x + 1)^3(3x - 1) + 1, \text{ and} \\ p_{3,2}(x) &= 2(x + 1)^2(3x^2 - 2x + 1) - 1 = 2x^3(3x + 4) + 1. \end{aligned}$$

This example is very comparable to the polynomial p_4 – in both settings, outside of the $x, x + 1$ factors that, we have a linear term and a quadratic term. So the probability that we have the necessary amount of smoothness from these factors will be the same as in the setting of p_4 (see Table 3 for concrete numbers).

Using the 64-bit twins that were found using CHM and setting $i = 3$ and $j = 2$, we best prime suitable for SQISign that was found is the following 253-bit prime, $p = 2(r + 1)^2(3r^2 - 2r + 1) - 1 = 2r^3(3r + 4) + 1$ with $r = 5964933197580566528$. Here we have

$$\begin{aligned} p + 1 &= 2 \cdot 3^5 \cdot 19 \cdot 31^2 \cdot 37^2 \cdot 67 \cdot 83^2 \cdot 89^2 \cdot 113^2 \cdot 157^4 \cdot 173^2 \cdot 233 \cdot 487^2 \\ &\quad \cdot 641 \cdot 70909698817 \cdot 293238280483753907, \text{ and} \\ p - 1 &= 2^{48} \cdot 11^3 \cdot 29^2 \cdot 47^3 \cdot 53^3 \cdot 79 \cdot 131^3 \cdot 331^3 \cdot 349^3 \cdot 439^3 \cdot 691 \\ &\quad \cdot 25693 \cdot 3792721. \end{aligned}$$

Using the 96-bit twins that were found using CHM and setting $i = 2$ and $j = 3$, we found the following 373-bit prime, $p = 2r^2(3r^2+8r+6)-1$ with $r = 64860388998155234831006433 27$. Here we have

$$\begin{aligned} p+1 &= 2 \cdot 3^7 \cdot 7^{10} \cdot 19^6 \cdot 67^2 \cdot 131 \cdot 241^2 \cdot 313^2 \cdot 379^2 \cdot 641 \cdot 883^2 \cdot 1103^2 \\ &\quad \cdot 1117^2 \cdot 2689 \cdot 11177 \cdot 1629009993156817 \\ &\quad \cdot 10232693625042911300232345793, \text{ and} \\ p-1 &= 2^{66} \cdot 5 \cdot 13^3 \cdot 17^3 \cdot 23^3 \cdot 41^3 \cdot 59^3 \cdot 61^3 \cdot 83^6 \cdot 127 \cdot 389 \cdot 491^3 \cdot 787^3 \\ &\quad \cdot 983 \cdot 1549^3 \cdot 10085189 \cdot 1986460467059. \end{aligned}$$

In addition, when reverting to $i = 3$ and $j = 2$, we found the following 380-bit prime $p = 2(r+1)^2(3r^2-2r+1)-1$ with $r = 2441295269140607126071436 9024$. This has an attractively small signing cost metric – the smallest among all of the all of our NIST-III primes while also attaining a relatively large power of two. Here we have

$$\begin{aligned} p+1 &= 2 \cdot 3^5 \cdot 5^4 \cdot 7^2 \cdot 17^2 \cdot 73 \cdot 79^4 \cdot 83^2 \cdot 179^2 \cdot 211^2 \cdot 239^2 \cdot 331^2 \cdot 353 \cdot 431^2 \\ &\quad \cdot 563 \cdot 1049^2 \cdot 1303^2 \cdot 1553 \cdot 2593 \cdot 12536326152153163 \\ &\quad \cdot 8137488798222954957535904 83, \text{ and} \\ p-1 &= 2^{57} \cdot 11^6 \cdot 19^3 \cdot 31^3 \cdot 41 \cdot 71^3 \cdot 89^3 \cdot 97^3 \cdot 101^3 \cdot 173^3 \cdot 191^3 \cdot 491^3 \\ &\quad \cdot 1301^3 \cdot 1523 \cdot 2932228515374750323580 83. \end{aligned}$$

Now we look at the case when $i, j \in \{2, 4\}$ with $i \neq j$. Here the degree of the polynomial $p_{i,j}(x)$ is 5 and, outside of the x^i and $(x+1)^j$ factors that appear in $p_{i,j}(x) \pm 1$, we have a linear and an irreducible cubic polynomial. So comparing it to the corresponding polynomial $p_5(x)$ where outside of x^5 and $x-1$ we only have an irreducible quartic, these polynomials offer a slightly better alternative to find SQISign parameters. While the amount guaranteed smoothness is smaller compared to the smaller i and j, the decomposition of the other factors made it possible to find some practical parameters. Using the 76-bit twins that were found using CHM and setting $i = 2$ and $j = 4$, we found the following 379-bit prime, $p = 2(r+1)^4(4r-1)+1$ with $r = 392939984404435717 32479$. Here we have

$$\begin{aligned} p+1 &= 2 \cdot 11^4 \cdot 13^2 \cdot 29^2 \cdot 53^2 \cdot 61^2 \cdot 127^2 \cdot 241^2 \cdot 277^2 \cdot 283^2 \cdot 293^2 \cdot 379^2 \\ &\quad \cdot 743 \cdot 907 \cdot 1289 \cdot 26417 \cdot 1356037 \cdot 103868239 \cdot 551183887 \\ &\quad \cdot 9479575745803 \cdot 14370319539299762939, \text{ and} \\ p-1 &= 2^{69} \cdot 3^{32} \cdot 5^5 \cdot 17^4 \cdot 23 \cdot 37^8 \cdot 41 \cdot 349^4 \cdot 409 \cdot 653^4 \cdot 839 \cdot 1367 \cdot 1723^4 \\ &\quad \cdot 10343 \cdot 6870751. \end{aligned}$$

For larger i and j, not only does the amount of guaranteed smoothness decrease but we are also hindered by the irreducibility of the polynomials $S_{i,j}$ and $T_{i,j}$. For instance using $i = 4$ and $j = 3$, one has to find smoothness from an irreducible quadratic and an irreducible cubic factor. In comparison to p_6, one has to look at a linear and two irreducible quadratic factors. This makes the smoothness probability from this choice of i, j worse. The experimentation with some of these larger i and j did not yield any practical SQISign parameters.

NIST security level	(i,j)	r	$\lceil \log_2(p) \rceil$	f	B	$\sqrt{B}/f$	$\log_p(T)$
NIST-I	$(3,1)$	27011253109821467755806711	245	35	673	0.74	1.28
		54021163652376016019456000	248	46	769	0.60	1.30
		55063253140688998251571200	248	31	457	0.69	1.35
		120028582508941249673379840	251	40	503	0.56	1.29
		313160539211983999706112000	256	37	491	0.60	1.29
	$(2,3)$	1661084743710662655	245	42	907	0.72	1.28
		5554268927024357375	252	42	547	0.56	1.26
	$(3,2)$	1254124477112410112	244	42	1103	0.79	1.32
		2066354955552768000	246	45	941	0.68	1.25
		5964933197580566528	253	48	691	0.55	1.29
NIST-III	$(2,3)$	6486038899815523483100643327	373	66	11177	1.60	1.26
		15243470554423552878739455999	378	57	85667	5.13	1.35
	$(3,2)$	17883472313040580309580840960	378	51	36913	3.77	1.30
		24412952691406071260714369024	380	57	2593	0.89	1.27
	$(2,4)$	16445101733510926196735	373	53	36061	3.58	1.31
		29470128945436815032319	377	61	36011	3.11	1.27
		30747138602455038607359	377	57	20441	2.51	1.29
		39293998440443571732479	379	69	26417	2.36	1.27
	$(4,2)$	6221427644839571619840	365	69	24631	2.27	1.27
		73986613900438892871680	383	61	40387	3.29	1.29

Table 6: A table of SQISign parameters $p = p_{i,j}(r)$ with $i, j \geq 2$ found using twin-smooth integers $(r, r+1)$ at each security level. The other quantities are just as in Table 4.

We report no NIST-V primes using this method. The experimentation using $i, j \in \{2, 3\}$ requires 128-bit twins in order obtain such primes. As mentioned before, the small quantity of such twins found from the CHM computation makes it probabilistically difficult to expect to find any SQISign parameters. Additionally, the experimentation for larger i, j did not result in any noteworthy primes that attained a large enough power of two while keeping the smoothness bound particularly small.

Remark VI.22. *We would ideally implement the primes from not only this section but also the primes found in Section VI.5.3 using the SQISign code provided in [DFLLW23] to determine how well these primes perform in practice. However, the current implementation is specifically tailored towards the particular primes that are being used, and is limited to NIST-I parameter sizes. Including our NIST-I primes from Table 4 results in failures during key generation, which seem to stem from using parameters with different powers of two. Thus, implementing and benchmarking our parameters would require a major rework of the provided code, which is out of the scope of this work. Nevertheless, the primes from Table 4 provide various alternatives for NIST-I parameters, and the first practical NIST-III and NIST-V parameters in the literature, which seems especially important in the light of the upcoming NIST submission period for signature schemes.*

Dropping the amount of guaranteed smoothness. The polynomials $p_{i,j}$ and their special cases p_n could guarantee a smooth factor $p^{1+1/n}$ where n is the degree of the polynomial. As demonstrated through Lemma VI.21, this is the largest one could guarantee from a smooth twin. So, for a fixed n, there are no other degree n polynomials that could guarantee either more or equal amount of smoothness from a single smooth twin. If we chose polynomials whereby the amount of guaranteed smoothness was smaller than this optimum, then our chances of finding suitable parameters for SQISign would be significantly smaller. However, one could counterbalance this by having a large family of polynomials that could guarantee a large amount of smoothness from a single twin. Subsequently, one could adopt some sieving techniques in order essentially test all options in one go. This would require using a combination of the following two sieving techniques. The first one is based on the sieve of Eratosthenes [CP01, Chapter 3.2.6] that identifies the smooth integers of the form $F(r)$ for a univariate polynomial F and from an interval of integers $r \in [a, b]$. The second one is due to Bernstein [Ber04] that identifies smooth integers from a list of numbers that has no prescribed structure. The idea is to compute the product of primes up to B, call this product z, then compute

$$y = (z \mod m)^{2^e} \mod m$$

for each integer m in the list where e is the smallest positive integer such that $2^{2^e} \geq m$ and finally compute gcd of y and m. This gcd computation gives the B-smooth part of the integer m [Ber04, Theorem 2.2] and moreover it is equal to m itself then the integer is B-smooth. Both techniques would be more than sufficient for the application to SQISign parameter searches of this type. No experimentation has conducted with this approach and we only theorise this as a possible avenue.

We go through this with a concrete example. Let a be a rational parameter and let $F(x) = x^2(ax + a + 1)$, $G(x) = x + 1$. Applying the extended Euclidean algorithm to these polynomials yields $S(x) = 1$ and $T(x) = -ax^2 - x + 1$. It is worth noting that, through a simple discriminant calculation, the polynomial T can be factored only when a is a product of consecutive integers. In this setting the smoothness probabilities will be slightly better due to the ability to factor T. Having said this it will be worthwhile to consider this for more values a other than this exceptional circumstance. Since $\deg(F \cdot S) = \deg(G \cdot T) = 3$, this would be comparable to the polynomial $p_3(x)$. Suppose that $(r, r + 1)$ are smooth twins. In comparison to p_3, the amount of guaranteed smoothness is reduced from around $p^{4/3}$ to around p/a^2. What we lose in guaranteed smoothness we make up for a parameter a. This means that from this one twin $(r, r + 1)$, we can try a test the smoothness of a lot more.

For each CHM smooth twin $(r, r + 1)$, let m_r be the polynomial $m_r(x) = (xr + x + 1)(xr^2 + r - 1)$. Then one would use the mentioned sieving techniques to sieve through this polynomial for integers inputs $a \in [1, N]$ and a given threshold N. This will tell us which values $m_r(a)$ have the necessary amount of smoothness for SQISign. Depending on the size of the threshold, the added cost of doing this is no longer negligible in comparison to the Section VI.5 and would most likely require the same amount or possibly more computation compared to the corresponding CHM computation.

Remark VI.23. *One could adopt this idea for a vast array of other settings other than just this exceptional case as long as the amount of guaranteed smoothness is quite large (say approximately p). If you adopted this for a family of polynomials of larger degree you would need to make sure that the polynomials s and t has some nice factoring properties which would certainly increase the smoothness probabilities. Depending on the polynomials in question, one would have to redo the CHM computations utilising the generalised CHM algorithm to find smooth values of other quadratic polynomials rather than just $x(x + 1)$. The new PTE solutions mentioned in Section VI.6.3, could be a good alternative to these polynomials.*

VI.8 Conclusion

In this chapter we have explored new techniques for finding twin-smooth integers. The first of these methods is based on the CHM algorithm which is a constructive algorithm that conjecturally finds “almost all" twin smooth integers for a given smoothness bound B. In particular we gave some optimisations to the algorithm in order to run the algorithm for larger values of B than previously done. The second of these methods is based on the extended Euclidean algorithm over the polynomial ring $\mathbb{Q}[x]$ and theoretically show when this method could be more advantageous over prior methods. We leave the practicalities of this algorithm and seeing whether one can find smoother cryptographic sized twins as future work.

Subsequently, we have put these techniques in the context of finding parameters to the isogeny-based signature scheme SQISign. The idea used to find these parameters is to find smaller twin-smooth integers and lift them up using either the polynomial $p_n(x) = 2x^n - 1$ or the polynomials $p_{i,j}(x)$ obtained from some XGCD computation.

As a final remark from the perspective of cryptographic protocols, other than digital signatures, there are no other known protocols based on the constructive Deuring correspondence. There seems to be a number of advanced protocols for which a SQISign adaptation could prove to be advantageous. Such a protocol (if possible) might require a setup that involves some twin smooth integers. We leave this exploration as an avenue for future work.

Chapter VII

Updatable Public-Key Encryption: Generic Constructions and Instantiations from Group Actions

In this chapter, we present work that was done in collaboration with Daniel Gardham [GS23].

VII.1 Introduction

Forward security prevents a compromised session from leaking information from previous sessions. That is, it enables users to regularly update keys so that any corruption at period i cannot help an adversary from breaking security for a time period $j > i$. In the public key setting, this is a trivial property if we allow the receiver to initiate the update, however, forward security instead allows a sender to initiate. This scenario is of particular interest to secure communication protocols, such as Signal [Sys22] or OTR [BGB04], where the receiver may be off line whilst receiving multiple messages from potentially multiple senders.

However, full forward secure PKE has been difficult to achieve with the best known constructions [JS18, PR18] making use of complex primitives such as Hierarchical Identity-based Encryption [HL02]. Thus, a weaker notion was considered: Updatable Public Key Encryption. This UPKE was initially proposed in Jost et. al. [JMM19] and swiftly built upon in Alwen et. al. [ACDT20], and relaxes full forward security by not requiring the need for the chain of public keys to be independently producible. Instead, a sender performing an update publishes the new public key on behalf of the receiver. Constructions for UPKE have been very efficient, that is, order of magnitude faster than comparable FS-PKE, which makes UPKE desirable in mobile devices for secure messaging protocols. However, there are few constructions of UPKE [JMM19, ACDT20, DKW21] all of which rely on specific hardness assumptions or underlying encryption schemes, for example, [DKW21] give a construction based on the BHHO cryptosystem. In particular, there is no generic approach for constructing this primitive despite advantages these methods can afford. By using generic building blocks, the schemes become completely independent from concrete assumptions, so that when a scheme is instantiated, one can employ the most efficient building blocks that meet the security guarantees and efficiency requirements defined by a particular deployment.

In addition, there is an ever growing need to build cryptographic primitives that are resistant to quantum adversaries. A vast majority of currently deployed public-key protocols rely on number-theoretic problems that will eventually be easy to solve when given a large enough quantum computer. Lattices provide a UPKE construction that is thought to be quantum-secure based on the Learning with Errors assumption [DKW21]. These constructions provide direct instantiations that rely on careful selection of parameters.

Isogenies have also been used to construct UPKE [EJKM22]. The first constructions was based on the SIDH framework but is now considered completely broken due to the recent attacks on the SIDH protocol [CD23, MMP+23, Rob23]. Their second construction is based on the isogeny-based group action which underpins the CSIDH protocol. The security of this scheme still remains intact however, as it shall be noted in the related work section, it requires an expensive pre-computation which makes it infeasible to realise with practical parameters.

VII.1.1 Contribution

In this chapter we give two new isogeny-based constructions of UPKE's based on a group action used in CSIDH. In particular the second of these constructions does not require this expensive pre-computation and hence makes it scalable. Moreover, these constructions are described using the language of cryptographic group actions.

In order to achieve this we first give generic constructions for UPKE that satisfy indistinguishability under a plaintext attack with chosen randomness, or IND-CR-CPA. We use key-private public-key encryption as a generic building block along with the ARKG protocol from Frymann et. al. [FGK+20]. We stress that the generic nature of our constructions allows the building blocks to be instantiated with any compatible KP-PKE and ARKG constructions. Secondly, we show that KP-PKE itself can be constructed in a generic way, once again utilising the ARKG protocol. When combined with standard public key encryption, we are able to construct IND-IK-CCA and IND-IK-CPA constructions for KP-PKE that are proven secure in the standard model. As a corollary to our first two contributions, we can construct UPKE from any compatible ARKG and PKE schemes using simplistic generic methods.

We then instantiate it with isogenies by giving two constructions for ARKG for both effective and restricted group actions. We show both meet all security properties of an ARKG scheme based on Group Action Inverse Problem, and a new assumption we call Psuedorandom Function Oracle Group Action Diffie Hellman (PRF-OGADH). This is based on an analogous assumption, PRF-ODH, that was introduced to study the security of TLS1.3 [BFGJ17] and underpins the original discrete-log based construction for ARKG. We believe this assumption may be of independent interest, and we show how it relates to the GAIP problem.

Combining our results thus far, we provide a concrete constructions of a UPKE from both effective and restricted effective group actions. Our construction based on restricted effective group actions gives a stronger alternative when instantiated with isogenies in comparison to the only known isogeny-based UPKE construction presented in [EJKM22].

VII.1.2 Related Works

Forward Secure Encryption. For public key encryption, Cannetti [CHK04] proposed the first scheme and remains state-of-the-art. The high-level idea is to define a key-pair $(\mathsf{sk}_i, \mathsf{pk}_i)$ for epoch i, where updating either pk or sk to the next epoch can be independently incremented by any, or possibly many, senders. Canetti also shows how to generically transform any Hierarchical Identity-based Encryption scheme into a FS-PKE. Since this result, many FS-PKE constructions have been constructed using this technique from a wide range of security assumptions such as DDH/CDH or factoring, or even LPN [DG17b, DG17a, BLSV18]. However these generic techniques have resulted in an efficiency gap between FS-PKE and standard PKE. Furthermore, all constructions that use the HIBE transform seem to require use of the Random Oracle Model. There are also lattice-based constructions for HIBE (e.g. [CHKP10]) which imply FS-PKE from lattices, whose security relies on the popular LWE assumption. However, this scheme suffers from inefficient GPV-style trapdoors [GPV08].

Updatable Public Key Encryption. As mentioned previously, UPKE is similar to FS-PKE but with a mild relaxation of the security properties. It was first defined in concurrent works of Alwen et. al. [ACDT20] and Jost et. al. [JMM19]. In these seminal works, Jost et. al. propose an efficient secure-messaging protocol with almost-optimal security in the setting where an adversary has access to intermediate values. They further consider randomness exposure, and rely on circular-security in the random oracle model to prove their protocol secure with these additional guarantees. Similarly, Alwen et. al. analyse the TreeKEM protocol [BBR18], which is at the core of the Secure Group Messaging (SGM) protocol proposed by the MLS working group [BRO$^+$22]. This paper gives a thorough analysis of a protocol they term *Continuous Group Key Agreement*, and show that TreeKEM falls short of required security. To overcome this, they propose a simpler definition for UPKE than that of Jost et. al , but realise it with the same construction based on the CDH assumption. This work also requires an assumption about the order in which messages are delivered to all participants.

Recently, Dodis et. al. [DKW21] improved upon prior work by giving the first UPKE schemes in the standard model. They further strengthen the security in a property they term Indistinguishability against Chosen-Randomness Plaintext Attacks, or IND-CR-CPA. It requires that exposure of any key secret key in epoch i does not compromise the messages encrypted under prior public keys in epoch j (where $j < i$), provided at least one update happened for which the adversary did not control the randomness used in the update. This is actually a stronger definition than originally proposed in Alwen et. al. [ACDT20] in which randomness is leaked to the adversary, but not controlled. Dodis et. al. give constructions based on a pairing based construction that relies on the BDDH assumption as well as a lattice construction based on LWE. They also consider and further strengthening of their security properties, that instead of allowing the adversary to choose the randomness, it can provide maliciously generated updated public keys and update tokens.

Finally we note that some works, for example [BLMR13, BDGJ20], called Updatable Encryption differ from UPKE since they have historically been based on symmetric encryption schemes. While some public-key constructions exist of Updatable Encryption scheme, the main difference is that the key pairs are updated together whereas in UPKE's the update of the public key and the private key's are done separately. This makes their use cases quite different. Hence, despite the similar name, they do not bare relation to Updatable Public Key Encryption.

Key-Private Encryption. In this work, it will be necessary to consider key-private public key encryption (KP-PKE). The defining property of KP-PKE ensures that an adversary is unable to distinguish which of two potential public keys were used to encrypt a ciphertext. This would be desired in situations where a receiver of a ciphertext wishes to remain anonymous on a shared channel. Along with UPKE, these anonymity properties of PKE have gained increasing importance in the anonymous communications space [BBDP01a].

Initially, it was shown that some well known encryption schemes such as ElGamal, Cramer-Shoup and RSA-based schemes have been shown to be key-private [BBDP01a]. Later, Paterson and Srinivasan [PS09], give the previously only generic approach to construct key-private PKE with IND-CCA security in a (multi-) trusted authority setting. Their techniques use the CHK transform [CHK04] and enjoy proofs in the standard model. It has been used by Kohlweiss et. al. [KMO$^+$], along with other standard PKE properties, to achieve confidential receiver-anonymous channels that preserve both message confidentiality and receiver anonymity, leaking only the length of the message and allowing the adversary only to delete, honestly deliver or inject arbitrary messages to chosen recipients. Finally, we also note that similar notions have been considered in identity-based encryption where key-privacy refers to the root authority. That is, an adversary must distinguish ciphertexts produced using different master public-keys, with control of the message and identity [PS08].

Cryptographic Group Actions. Over the last few years, the topic of cryptographic group actions has gained a lot of attraction. These group actions can be thought of as a generalisation of the traditional group exponentiation that is found in a lot of modern protocols. As already mentioned in Chapter IV the class group of an order in an imaginary quadratic field acts on the set of rational supersingular elliptic curves by isogenies. This gives us the CSIDH key exchange protocol and can be reformulated as a cryptographic group action. An updatable public key encryption scheme was proposed by Eaton et. al. [EJKM22] using this specific group action. However, it requires an expensive class group precomputation in order to run their protocol which, as mentioned in Section IV.5.1, has only been done for certain parameters and does not scale for the higher security levels. However, as part of recent work by De Feo et. al. [FFK$^+$23], were able to scale this by working with a quadratic order of a specific description which makes computing the class group more efficient. Nonetheless, the cost of evaluating the group action still has subexponential complexity and makes it rather impractical. In addition, due to recent quantum security analysis of these group actions [Pei20, BS20], the parameters that were initially proposed fall short of their claimed security targets. This would therefore require the size of the parameters to increase by quite a margin [CSCDJRH22]. This would make the precomputation task completely infeasible. The CSIDH non-interactive key exchange protocol [CLM$^+$18] gets around this by having a specific parameter selection and uses a set of known ideal classes that at the very least is conjectured to cover the class group but means that one cannot guarantee absolute uniform sampling within the whole group.

In 2020, the idea of cryptographic group actions was formalised by Alamati et al. [ADFMP20]. This formalisation captures not only the general setting of a cryptographic group action (which they call *effective* group actions) but also captures the restricted case that encompasses the setup given in CSIDH (which they call *restricted effective* group actions). From this formalisation they were able to construct dual-mode PKE and a Naor-Reingold style PRF (among other things). This formalisation has been explored by others to build other protocols such as password authenticated key exchange and key encapsulation mechanisms [AEK+22, DHK+23]. Recently, Leroux and Roméas [LR22] made some strides to construct an updatable encryption scheme from group actions. As mentioned earlier, this scheme bares no relation to the updatable public key encryption that will be the focus of this work.

VII.1.3 Organisation

In Section VII.2 we introduce the cryptographic tools and preliminaries. This includes a formal description of cryptographic group actions and their underlying hardness assumptions as well as the cryptographic building blocks that will be needed for our purposes. In Section VII.3 we prove the generic construction of a UPKE scheme that uses the functionality of an ARKG scheme as well as a KP-PKE scheme. In Section VII.4 we prove a generic construction of the KP-PKE from an ARKG scheme as well as a PKE scheme. From this, one can conclude that the UPKE construction can be obtained from an ARKG scheme and a PKE scheme. In Section VII.5, we use the generic techniques to provide a concrete construction of the UPKE that uses the language of cryptographic group actions. In the Appendices, we provide some extra reductions and security proofs.

VII.2 Preliminaries

We begin by introducing the underlying mathematical concepts and cryptographic building blocks.

VII.2.1 Cryptographic Group Actions

Given a group G and a set X, we recall that we say G acts on X by a map $\star : G \times X \to X$ if this map satisfies the following two properties:

- If e is the identity element in G, then $e \star x = x$ for any $x \in X$;
- $g \star (h \star x) = (gh) \star x$ for any $g, h \in G$ and $x \in X$.

Whenever this is the case, we say the triple $(G, X, \star)$ is a group action (often we abbreviate this to just referring to $\star$ as group action). There is a notion of a cryptographic group action on which the problem of recovering a group element g when presented with x and $g \star x$ is hard. Stolbunov [RS06] called this problem the Group Action Inverse Problem (GAIP). In recent years, more formal definitions of these cryptographic group actions have been presented that give a rigorous reflection on what one requires for these group actions. This formalisation was done by Alamati et.al. [ADFMP20] and we adopt their definition in this work.

Definition VII.1 (Effective group actions (EGA))**.** *We say a group action $(G,X,\star)$ is effective if the following properties are satisfied*

1. *The group G is finite and there are efficient (PPT) algorithms for:*
   - *Membership testing: decide whether an element g belongs to the group G;*
   - *Equality testing: decide whether two elements, $g_0, g_1 \in G$ represent the same group element in G;*
   - *Sampling: from a distribution $\mathscr{D}_G$ on G, sample elements from G;*
   - *Operation: computing gh for any $g,h \in G$;*
   - *Inversion: computing g^{-1} for any $g \in G$.*
2. *The set X is finite and there are efficient algorithms for:*
   - *Membership testing: decide whether an element x belongs to the set X;*
   - *Unique representation: given an arbitrary set element $x \in X$, compute a string $\hat{x}$ that canonically represents x.*
3. *There is a distinguished element $x_0 \in X$, called the "origin", such that its bit-string representation is known.*
4. *There is an efficient algorithm that given any $g \in G$ and $x \in X$ outputs $g \star x$.*

Next we introduce a slightly modified definition of *restricted effective group actions*. The slight modification encompasses the fact that sampling elements from the underlying generating set is close to uniform up to some negligible probability.

Definition VII.2 (Restricted effective group actions (REGA))**.** *Let $(G,X,\star)$ be a group action and let $\mathbf{g} = (g_1, \cdots, g_n)$ be a (not necessarily minimal) generating set for G. We say the action is $\mathbf{g}$-restricted effective if the following properties are satisfied*

1. *G is finite and $n = \mathrm{poly}(\log(|G|)$.*
2. *The set X is finite and there are efficient algorithms for:*
   - *Membership testing: decide whether an element x belongs to the set X;*
   - *Unique representation: given an arbitrary set element $x \in X$, compute a string $\hat{x}$ that canonically represents x.*
3. *There is a distinguished element $x_0 \in X$, called the "origin", such that its bit-string representation is known.*
4. *There is an efficient algorithm that given any $g_i \in \mathbf{g}$ in the generating set for G and $x \in X$ outputs $g_i \star x$ and $g_i^{-1} \star x$.*

Remark VII.3. *As described in Section IV.5, the group action based on isogenies that is used in CSIDH can be turned into a REGA.*

Remark VII.4. *For a REGA, if the generators g_i are uniformly random elements of G, then the resulting distribution of elements of the form $\prod g_i^{e_i}$ is statistically close to uniform. In the setting of CSIDH, this argument can only be made heuristically.*

We abuse the naming conventions given to these types of group actions by calling these cryptographic group actions whenever we are in at least one of either of these definitions. We formalise the *group action inverse problem* (GAIP) which is a core and fundamental problem for these cryptographic group actions that we want to be hard.

Definition VII.5 (Group Action Inverse Problem)**.** *Let $(G, X, \star)$ be a cryptographic group action with a distinguished element $x \in X$ and let $y := g \star x$ for some $g \in G$. The "group action inverse problem" (GAIP) asks to compute the element g given the set elements x, y.*

In the formalisation of these group actions by Alamati et. al. [ADFMP20], they introduced a security definition which they call weak-unpredicatable group action. The idea that captures this definition is that any polynomial time adversary cannot compute $g \star x^*$ for some challenge $x^* \in X$ even if it is given polynomially many tuples of the form $(x_i, g \star x_i)$ with $x_i \neq x^*$.

However, in this work we adopt the definitions laid out by Abdalla et. al. [AEK+22] which are a more traditional formulation of the security properties that are desired for these cryptographic group actions. Moreover, these definitions are equivalent to the weak-unpredictable definition mentioned above. We recall the main definitions that are needed for this work.

Definition VII.6 (Group action decisional Diffie Hellman Problem)**.** *Let $(G, X, \star)$ be a cryptographic group action (either effective or restricted) with a distinguished element $x \in X$ and $g \star x, h \star x, y \in X$ be three set elements. The group action decision oracle* $\mathrm{GA-DDH}$ *takes as input the three set elements and returns either 1 if $y = (g \cdot h) \star x$ or 0 otherwise. Moreover we define the* $\mathrm{GA-DDH}$ *advantage for an adversary $\mathscr{A}$ to be*

$$\mathrm{Adv}^{\mathrm{GA-DDH}}_{(G,X,\star),\mathscr{A}}(\lambda) := \Pr\left[\mathscr{A}(g \star x, h \star x, y) = b' : b' = \mathrm{GA-DDH}(g \star x, h \star x, y)\right].$$

Definition VII.7 (Group action strong computational Diffie Hellman Problem)**.** *Let $(G, X, \star)$ be a cryptographic group action (either effective or restricted) with a distinguished element $x \in X$ and $g \star x, h \star x \in X$ be two set elements. The problem to compute the set element $(g \cdot h) \star x$ given only $g \star x$ and $h \star x$ as well as access to a decisional oracle. Moreover we define the* $\mathrm{GA-StCDH}$ *advantage for an adversary $\mathscr{A}$ to be*

$$\mathrm{Adv}^{\mathrm{GA-StCDH}}_{(G,X,\star),\mathscr{A}}(\lambda) := \Pr\left[\mathscr{A}^{\mathrm{GA-DDH}(g \star x, \cdot, \cdot)}(g \star x, h \star x) = (g \cdot h) \star x\right].$$

VII.2.2 Cryptographic Building Blocks

In this section we recall the definitions and security properties of standard cryptographic primitives.

Asynchronous Remote Key Generation [FGK+20]. This building block enables for asynchronous key generation. That is, the creation of a public key via the probabilistic algorithm DerivePK can occur before that of the corresponding secret key, via the deterministic algorithm DeriveSK. This will form the update procedure of our UPKE constructions. Initially, it was motivated as protocol for WebAuthn account recovery, although the critical property that the public key generation can be separated from the secret key generation can be used in other applications, as we do in this work. The high-level description of ARKG follows.

The Asynchronous Remote Key Generation scheme consists of the following five algorithms $\mathsf{ARKG} := (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{DerivePK}, \mathsf{DeriveSK}, \mathsf{Check})$:

$\mathsf{Setup}(1^\lambda)$ generates and outputs public parameters of the scheme for the security parameter $\lambda \in \mathbb{N}$.

$\mathsf{KeyGen}(\mathsf{pp})$, on input pp, computes and returns a private-public key pair $(\mathsf{sk}, \mathsf{pk})$.

$\mathsf{DerivePK}(\mathsf{pp}, \mathsf{pk}, \mathsf{aux})$ probabilistically returns a new public key pk' together with the link cred between pk and pk', for the inputs pp, pk and auxiliary data aux. In this work, we do not make use of aux and therefore do not include it in algorithm calls.

$\mathsf{DeriveSK}(\mathsf{pp}, \mathsf{sk}, \mathsf{cred})$ is a deterministic algorithm that computes and outputs either the new private key sk', corresponding to the public key pk' using cred, or $\perp$ on error. Together, $(\mathsf{sk}', \mathsf{pk}')$ have a distribution we call $\mathscr{D}$.

$\mathsf{Check}(\mathsf{pp}, \mathsf{sk}', \mathsf{pk}')$, on input $(\mathsf{sk}', \mathsf{pk}')$, returns 1 if $(\mathsf{sk}', \mathsf{pk}')$ forms a valid private-public key pair, where sk' is the corresponding private key to pk', else 0.

Correctness. An ARKG scheme is correct if, for every $\lambda \in \mathbb{N}$, $\mathsf{pp} \leftarrow \mathsf{Setup}(\lambda)$, the probability $\Pr\big[\mathsf{Check}(\mathsf{pp}, \mathsf{sk}', \mathsf{pk}') = 1\big] = 1$ given

$$\begin{aligned} (\mathsf{sk}, \mathsf{pk}) &\leftarrow \mathsf{KeyGen}(\mathsf{pp}); \\ (\mathsf{pk}', \mathsf{cred}) &\leftarrow \mathsf{DerivePK}(\mathsf{pp}, \mathsf{pk}, \cdot); \\ \mathsf{sk}' &\leftarrow \mathsf{DeriveSK}(\mathsf{pp}, \mathsf{sk}, \mathsf{cred}). \end{aligned}$$

We recall the security properties for ARKG. The first, PK Unlinkability, is an anonymity property that ensures derived keys cannot be linked to the long term key, whereas key-forgery is prevented by SK security. The latter comes in 4 variations depending on the power afforded to the adversary. In this work we implicitly make use of the weakest definition, Honest Strong SK security (HSKS). We further introduce a new property for ARKG that requires existence of an algorithm that can simulate derivation but for a preselected 'derived' public key.

SK-security. The private-key security property ensures that for an initial public key pk, an adversary $\mathscr{A}$ cannot derive a valid key pair $(\mathsf{sk}^\star, \mathsf{pk}^\star)$ along with corresponding $\mathsf{cred}^\star$. We consider the range of variants of private-key security defined in Fryman et al., modelled using the experiment $\mathsf{Exp}^{\mathsf{ks}}_{\mathsf{ARKG},\mathscr{A}}(\lambda)$ in fig. 12 with $\mathsf{ks} \in \{\mathsf{mwKS}, \mathsf{hwKS}, \mathsf{msKS}, \mathsf{hsKS}\}$. Adversary $\mathscr{A}$ is always given access to $\mathscr{O}_{\mathsf{pk}'}$ and must find a $(\mathsf{sk}^\star, \mathsf{pk}^\star, \mathsf{cred}^\star)$ triple for a provided pk.

The malicious (m) and honest (h) variants result from the omission or presence of the PKList check on line 8, respectively, which ensures that the triple is for an honestly-generated pk (modelled using $\mathscr{O}_{\mathsf{pk}'}$) if present. The weak (w) and strong (s) variants depend on whether $\mathscr{A}$ has access to the private key derivation oracle $\mathscr{O}_{\mathsf{sk}'}$. If $\mathscr{A}$ has access to $\mathscr{O}_{\mathsf{sk}'}$, trivially querying it with $\mathsf{cred}^\star$ is prevented through the SKList check on line 7.

```
$\mathsf{Exp}^{\mathsf{ks}}_{\mathsf{ARKG},\mathcal{A}}(\lambda)$
1 : $\mathsf{pp} \leftarrow \mathsf{Setup}(\lambda)$
2 : $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KeyGen}(\mathsf{pp})$
3 : $(\mathsf{sk}^\star, \mathsf{pk}^\star, \mathsf{cred}^\star) \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{pk}'}, \boxed{\mathcal{O}_{\mathsf{sk}'}}} t(\mathsf{pp}, \mathsf{pk})$
4 : $\mathsf{sk}' \leftarrow \mathsf{DeriveSK}(\mathsf{pp}, \mathsf{sk}, \mathsf{cred}^\star)$
5 : return $\mathsf{Check}(\mathsf{sk}^\star, \mathsf{pk}^\star) \overset{?}{=} 1$
6 :         $\land\ \mathsf{Check}(\mathsf{sk}', \mathsf{pk}^\star) \overset{?}{=} 1$
7 :         $\boxed{\land\ \mathsf{cred}^\star \notin \mathsf{SKList}}$
8 :         $\boxed{\land\ (\mathsf{pk}^\star, \mathsf{cred}^\star) \in \mathsf{PKList}}$
```

```
$\mathcal{O}_{\mathsf{pk}'}(\mathsf{aux})$
1 : $\mathsf{pk}', \mathsf{cred} \leftarrow \mathsf{DerivePK}(\mathsf{pp}, \mathsf{pk}, \mathsf{aux})$
2 : $\mathsf{PKList} \leftarrow \mathsf{PKList} \cup \{(\mathsf{pk}', \mathsf{cred})\}$
3 : return $\mathsf{pk}', \mathsf{cred}$
```

```
$\mathcal{O}_{\mathsf{sk}'}(\mathsf{cred})$
1 : if $\{(\cdot, \mathsf{cred})\} \notin \mathsf{PKList}$ then
2 :     return $\perp$
3 : $\mathsf{sk}' \leftarrow \mathsf{DeriveSK}(\mathsf{pp}, \mathsf{sk}, \mathsf{cred})$
4 : $\mathsf{SKList} \leftarrow \mathsf{SKList} \cup \{(\mathsf{sk}', \mathsf{cred})\}$
5 : return $\mathsf{sk}'$
```

Figure 11: Key secrecy experiment and oracle description for ARKG. *The dotted boxes describe the four variants of the* $ks \in \{\mathsf{mwKS}, \mathsf{hwKS}, \mathsf{msKS}, \mathsf{hsKS}\}$ *experiment. Presence of the dashed boxes gives the strong variants* msKS *and* hsKS, *the presence of the dotted box gives the honest variants* hwKS *and* hsKS, *and the omission of all boxes gives* mwKS.

Definition VII.8 (SK-security). *An* ARKG *scheme provides private-key security for the different variants,* $ks \in \{\mathsf{mwKS}, \mathsf{hwKS}, \mathsf{msKS}, \mathsf{hsKS}\}$, *if the following advantage is negligible in* λ*:*

$$\mathsf{Adv}^{ks}_{\mathsf{ARKG},\mathcal{A}}(\lambda) := \Pr\Big[\mathsf{Exp}^{ks}_{\mathsf{ARKG},\mathcal{A}}(\lambda) = 1\Big]$$

PK-unlinkability. This property ensures that derived key pairs cannot be distinguished from a sample of a distribution $\mathcal{D}$ and also prevents an adversary from linking a derived public key to a long-term public key. The unlinkability between public keys pk and derived public keys pk$'$ is defined using $\mathsf{Exp}^{\mathsf{PKU}}_{\mathsf{ARKG},\mathcal{A}}(\lambda)$. The game generates a key pair $(\mathsf{sk}_0, \mathsf{pk}_0)$ and flips a bit b. Then, $\mathcal{A}$ is given access to oracle $\mathcal{O}^b_{\mathsf{pk}'}$, public parameters pp, and pk_0. When the oracle is called, it returns a derived key pair $(\mathsf{sk}', \mathsf{pk}')$, which is derived from pk_0 if $b = 0$, otherwise, for $b = 1$, it samples and returns key pair $(\mathsf{sk}', \mathsf{pk}')$ according to a distribution $\mathcal{D}$. It is able to corrupt any derived key, for which corresponding sk$'$ is output. The adversary $\mathcal{A}$ wins the game if it is able to determine whether $\mathcal{O}^b_{\mathsf{pk}'}$ is instantiated with $b = 0$ or $b = 1$.

Definition VII.9 (PK-unlinkability). *An* ARKG *scheme provides PK-unlinkability if the following advantage is negligible in* λ*:*

$$\mathsf{Adv}^{\mathsf{PKU}}_{\mathsf{ARKG},\mathcal{A}} := \left|\Pr\Big[\mathsf{Exp}^{\mathsf{PKU}}_{\mathsf{ARKG},\mathcal{A}}(\lambda) = 1\Big] - \frac{1}{2}\right|$$

Simulatable Key Derivation. We introduce a new security property that allows a party falsify a derivation by *a priori* selecting the derived public key, and computing the delegator's key and randomness needed to derive it. We say that an ARKG scheme has Simulatable Public Key Derivation if there exists an algorithm T-Derive (pk$'$) takes input a "derived" [8] public key pk$'$ and outputs pk and r such that $\mathsf{pk}' \leftarrow \mathsf{DerivePK}(\mathsf{pk}; r)$. The semi-colon notation means

[8]We call this a derived public key since it plays the role of pk$'$ in ARKG, however we stress this can be any adversarially chosen public key.

```
$\mathsf{Exp}^{\mathsf{PKU}}_{\mathsf{ARKG},\mathscr{A}}(\lambda)$
1 : $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$
2 : $(\mathsf{sk}_0, \mathsf{pk}_0) \leftarrow \mathsf{KeyGen}(\mathsf{pp})$
3 : $b \leftarrow\$ \{0,1\}$
4 : $b' \leftarrow \mathscr{A}^{\mathscr{O}^b_{\mathsf{pk}'}}(\mathsf{pp}, \mathsf{pk}_0)$
5 : return $b \overset{?}{=} b'$
```

```
$\mathscr{O}^b_{\mathsf{pk}'}(\mathsf{sk}_0, \mathsf{pk}_0)$
1 : $(\mathsf{sk}'_1, \mathsf{pk}'_1) \leftarrow \$\mathscr{D}$
2 : $\mathsf{pk}'_0, \mathsf{cred} \leftarrow \mathsf{DerivePK}(\mathsf{pp}, \mathsf{pk}_b, \mathsf{aux})$
3 : $\mathsf{sk}'_0 \leftarrow \mathsf{DeriveSK}(\mathsf{pp}, \mathsf{sk}_b, \mathsf{cred})$
4 : return $(\mathsf{sk}'_b, \mathsf{pk}'_b)$
```

Figure 12: PK Unlinkability experiment and oracle description for ARKG.

we have fixed the randomness for this algorithm with the value r. Note that we only require T-Derive to output one instance of the pair (pk, r), i.e. it can only be called on pk' once. This property is inherent in the construction of Frymann et al. [FGK+20], in the Random Oracle Model. The motivation behind this property is to facilitate the security proofs of the generic UPKE construction.

Key-Private Public-Key Encryption. This is defined as public key encryption, but with the additional property, key-privacy. Intuitively, this prevents an adversary from linking a ciphertext to the public-key it was encrypted with. The formal definition is given in Definition VII.10. We note that many popular encryption schemes such as Elgamal and Cramer-Shoup naturally satisfy this property without modification [BBDP01a].

We capture the key privacy property in the indistinguishability of keys experiments formally defined in Figure 13, and modelled from Paterson et al. [PS09]. Intuitively, the experiment requires a two stage adversary $\mathscr{A} = (\mathscr{A}_1, \mathscr{A}_2)$ to identify which of two public key and message pairs created a ciphertext. The adversary submits challenge messages m_0 and m_1 that will be encrypted to form the challenge ciphertext under pk_0 and pk_1 respectively. In the IND-CCA version of the experiment, it also has access to a decryption oracle that will decrypt ciphertexts using secret key according to the adversary's input b^*. Note that this oracle will abort if the decryption fails (such as if the secret key selected does not match the ciphertext). It wins if it can correctly guess which of the pairs $(\mathsf{pk}_0, \mathsf{m}_0)$ or $(\mathsf{pk}_1, \mathsf{m}_1)$ was used to create the challenge ciphertext.

Definition VII.10. *A* KP-PKE *scheme said to be* $\mathsf{atk} \in \{\mathsf{IND\text{-}IK\text{-}CPA}, \mathsf{IND\text{-}IK\text{-}CCA}\}$ *secure if the following advantage is negligible in the security parameter* λ.

$$\mathsf{Adv}^{\mathsf{atk}}_{\mathscr{A},\mathsf{KP\text{-}PKE}}(\lambda) := \left|\Pr\left[\mathsf{Exp}^{\mathsf{atk}-1}_{\mathscr{A},\mathsf{KP\text{-}PKE}}(\lambda) = 1\right] - \Pr\left[\mathsf{Exp}^{\mathsf{atk}-0}_{\mathscr{A},\mathsf{KP\text{-}PKE}}(\lambda) = 1\right]\right|$$

Updatable Public-Key Encryption. We use the strong definition for UPKE from [DKW21]. It consists of five algorithms, that separate out the update procedures from the encryption and decryption algorithms. As for any public-key cryptographic protocol that is somehow time dependent, we keep track of this with epochs. For UPKE, the epoch is advanced when a new an updated public key is computed. The algorithms are as follows.

KeyGen (1^λ) On input of a security parameter returns an initial key-pair $(\mathsf{sk}_0, \mathsf{pk}_0)$.

```
$\mathsf{IND\text{-}IK\text{-}CPA} - b$
1 : $(\mathsf{sk}_0, \mathsf{pk}_0) \leftarrow \mathsf{KeyGen}(1^\lambda)$
2 : $(\mathsf{sk}_1, \mathsf{pk}_1) \leftarrow \mathsf{KeyGen}(1^\lambda)$
3 : $(\mathsf{m}_0, \mathsf{m}_1, \mathsf{st}) \leftarrow \mathscr{A}_1(\mathsf{pk}_0, \mathsf{pk}_1)$
4 : $\mathsf{c}^* \leftarrow \mathsf{Encrypt}(\mathsf{pk}_b, \mathsf{m}_b)$
5 : $b' \leftarrow \mathscr{A}_2(\mathsf{c}^*, \mathsf{st})$
6 : return $b \overset{?}{=} b'$
```

```
$\mathsf{IND\text{-}IK\text{-}CCA} - b$
1 : $(\mathsf{sk}_0, \mathsf{pk}_0) \leftarrow \mathsf{KeyGen}(1^\lambda)$
2 : $(\mathsf{sk}_1, \mathsf{pk}_1) \leftarrow \mathsf{KeyGen}(1^\lambda)$
3 : $(\mathsf{m}_0, \mathsf{m}_1, \mathsf{st}) \leftarrow \mathscr{A}_1^{\mathscr{O}_{\mathsf{dec}}}(\mathsf{pk}_0, \mathsf{pk}_1)$
4 : $\mathsf{c}^* \leftarrow \mathsf{Encrypt}(\mathsf{pk}_b, \mathsf{m}_b)$
5 : $b' \leftarrow \mathscr{A}_2(\mathsf{c}^*, \mathsf{st})$
6 : return $b \overset{?}{=} b'$
```

```
$\mathscr{O}_{\mathsf{dec}}(b^*, \mathsf{c})$
1 : $\mathsf{m} \leftarrow \mathsf{Decrypt}(\mathsf{sk}_{b^*}, \mathsf{c})$
2 : return $\mathsf{m}$
```

Figure 13: Security experiments and respective oracles for Key Privacy.

$\mathsf{UpdatePK}\ (\mathsf{pk}_i)$ Takes as input a public key pk_i and outputs an updated public key pk_{i+1} and update ciphertext up_{i+1}.

$\mathsf{UpdateSK}\ (\mathsf{sk}_i, \mathsf{up}_{i+1})$ Takes as input a secret key sk_i and update ciphertext up_{i+1} and returns updated secret key sk_{i+1}.

$\mathsf{Encrypt}\ (\mathsf{pk}_i, \mathsf{m})$ computes a ciphertext c from the input message m and public key pk_i.

$\mathsf{Decrypt}\ (\mathsf{sk}_i, \mathsf{c})$ From input of a ciphertext c and secret key sk_i, returns message m.

Correctness. Let $(\mathsf{pk}_i, \mathsf{up}) \leftarrow \mathsf{UpdatePK}(\mathsf{pk}), (\mathsf{sk}_i) \leftarrow \mathsf{UpdateSK}(\mathsf{sk}, \mathsf{up})$, then a UPKE scheme is correct if, for any message m, the following holds:

$$\Pr\big[\mathsf{Decrypt}(\mathsf{sk}_i, \mathsf{Encrypt}(\mathsf{pk}_i, \mathsf{m})) = \mathsf{m}\big] = 1.$$

Security. We consider a strong definition of security for UPKE based on IND-CPA for PKE. It is introduced by Dodis et al. [DKW21] and is called Indistinguishability against chosen plaintext attacks with chosen randomness, or IND-CR-CPA. Intuitively, it follows the standard definition of IND-CPA but the adversary can control the randomness used to update keys. It has access to an oracle $\mathscr{O}_{\mathsf{upd}}$ that computes derived public keys with randomness r submitted by the adversary. This oracle progresses the epoch each time it is queried by incrementing i, initially set to 0. When the challenge messages are selected, the experiment flips a bit and encrypts the corresponding message. This ciphertext is passed to the adversary as the challenge ciphertext. Then, the experiment updates the challenge keys once more with randomness not selected by the adversary. It is then given the derived public key, corresponding secret key and update token, and must guess the value of the bit b. A UPKE scheme is IND-CR-CPA secure if the adversary has negligible advantage in winning this experiment. We present the formal definitions in Definition VII.11.

Definition VII.11. *A* UPKE *scheme said to be* $\mathsf{IND\text{-}CR\text{-}CPA}$ *secure if the following advantage is negligible in the security parameter* λ.

```
IND-CR-CPA − $b$
1 : $(\mathsf{sk}_0, \mathsf{pk}_0) \leftarrow \mathsf{KeyGen}(1^\lambda)$
2 : $(\mathsf{m}^{(0)}, \mathsf{m}^{(1)}, \mathsf{st}) \leftarrow \mathscr{A}^{\mathcal{O}_{\mathsf{upd}}}(\mathsf{pk}_0)$
3 : $\mathsf{c}^* \leftarrow \mathsf{Encrypt}(\mathsf{pk}_i, \mathsf{m}^{(b)})$
4 : $\mathsf{st} \leftarrow \mathscr{A}^{\mathcal{O}_{\mathsf{upd}}}(\mathsf{c}^*, \mathsf{st})$
5 : $(\mathsf{up}^*, \mathsf{pk}^*) \leftarrow \mathsf{UpdatePK}(\mathsf{pk}_i)$
6 : $\mathsf{sk}^* \leftarrow \mathsf{UpdateSK}(\mathsf{sk}_i, \mathsf{up}^*)$
7 : $b' \leftarrow \mathscr{A}(\mathsf{pk}^*, \mathsf{sk}^*, \mathsf{up}^*, \mathsf{st})$
8 : return $b \overset{?}{=} b'$
```

```
$\mathcal{O}_{\mathsf{upd}}(r)$
1 : Retrieve $\{\mathsf{sk}_i, \mathsf{pk}_i\}$ from $\mathsf{KL}$
2 : $(\mathsf{pk}_{i+1}, \mathsf{up}_{i+1}) \leftarrow \mathsf{UpdatePK}(\mathsf{pk}_{i+1}; r)$
3 : $\mathsf{sk}_{i+1} \leftarrow \mathsf{UpdateSK}(\mathsf{sk}_{i+1}, \mathsf{up}_{i+1})$
4 : $\mathsf{KL} \leftarrow \mathsf{KL} \cup \{\mathsf{sk}_{i+1}, \mathsf{pk}_{i+1}, \mathsf{up}_{i+1}\}$
5 : $i = i + 1$
6 : return $\mathsf{pk}_i$
```

Figure 14: Security experiments and respective oracles for UPKE.

$$\mathsf{Adv}^{\mathsf{atk}}_{\mathscr{A},\mathsf{UPKE}}(\lambda) := \left|\Pr\left[\mathsf{Exp}^{\mathsf{IND\text{-}CR\text{-}CPA}-1}_{\mathscr{A},\mathsf{UPKE}}(\lambda) = 1\right] - \Pr\left[\mathsf{Exp}^{\mathsf{IND\text{-}CR\text{-}CPA}-0}_{\mathscr{A},\mathsf{UPKE}}(\lambda) = 1\right]\right|$$

Pseudorandom Function (PRF) ***[GGM86].*** A pseudorandom function $\mathsf{PRF}(k, m)$ is a family of deterministic functions indexed by a key k and message m and produces an output indistinguishable from a uniformly-sampled output for PPT adversaries. The adversary has access to oracle $\mathcal{O}_{\mathsf{PRF}}(\cdot)$ which cannot be queried with m and returns either $\mathsf{PRF}(k, \cdot)$ or $f(\cdot)$, with f being a truly random function.

Key Derivation Function (KDF) ***[Kra10].*** A key derivation function $\mathsf{KDF}(k, l)$ takes a key k and label l and returns a new key k'. We say it is secure if the advantage $\mathsf{Adv}^{\mathsf{KDF}}_{\mathscr{A}}(\lambda)$ is negligible in λ for a PPT adversary $\mathscr{A}$ to distinguish derived keys from uniformly-sampled elements in the output domain. We denote $\mathsf{KDF}_1(k) = \mathsf{KDF}(k, l_1)$ and $\mathsf{KDF}_2(k) = \mathsf{KDF}(k, l_2)$, where l_1, l_2 are implicit fixed labels. In particular, we often drop the labels from the input.

Message Authentication Code (MAC) ***[BCK96].*** A message authentication code $\mathsf{MAC} = (\mathsf{KeyGen}, \mathsf{Tag}, \mathsf{Verify})$ consists of three algorithms. $\mathsf{KeyGen}(1^\lambda)$ outputs secret key $mk \leftarrow\$ \{0,1\}^\lambda$ for a given security parameter λ. $\mathsf{Tag}(mk, m)$ outputs tag μ for input key mk and message m, and $\mathsf{Verify}(mk, m, \mu)$ outputs 1 if μ is valid for m under mk, otherwise 0. The correctness property is satisfied if $\forall(mk, m)$, $\mathsf{Verify}(mk, m, \mathsf{Tag}(mk, m)) = 1$. We say that MAC is unforgeable if the advantage $\mathsf{Adv}^{\mathsf{MAC}}_{\mathscr{A}}(\lambda)$ is negligible in λ for a PPT adversary $\mathscr{A}$ to find, without mk, a valid tag $\mu^\star$ for a new message $m^\star$. $\mathscr{A}$ is given access to oracle $\mathcal{O}_{\mathsf{Tag}}(\cdot)$, which on input message $m \neq m^\star$ returns the result of $\mathsf{Tag}(mk, m)$.

VII.3 Generic Construction for Updatable Public Key Encryption

We now present our generic construction for UPKE. We use an asynchronous remote key generation scheme alongside a key-private public key encryption scheme to create a UPKE. ARKG is used to enable the key update feature of our UPKE. On the other hand, we require key-private PKE to facilitate the security proofs. We note that there needs to be some compatibility between

ARKG and KP-PKE that are used in order to make the UPKE construction. This is because outputs of the ARKG primitive are used as keys in the KP-PKE primitive. Hence one cannot use any construction of an ARKG or KP-PKE (which may rely on vastly different assumptions) together to give a UPKE. However, for the purpose of this exposition, we assume that there is this compatibility and proceed with a high-level description of our protocol. The full algorithms are given in Figure 15.

$\mathsf{Setup}\ (1^\lambda)$ Setup takes as input a security parameter and executes the setup algorithms for both underlying building blocks ARKG and KP-PKE. It outputs the result in public parameters pp.

$\mathsf{KeyGen}\ (\mathsf{pp})$ The key generation algorithm for UPKE takes as input pp and runs $\mathsf{ARKG.KeyGen}$. It outputs the initial keys for the scheme as $(\mathsf{sk}_0, \mathsf{pk}_0)$.

$\mathsf{UpdatePK}\ (\mathsf{pp}, \mathsf{pk}_i)$ This algorithm takes as input a public key pk_i for epoch i and pp. It runs $\mathsf{ARKG.DerivePK}$, which outputs the new public key pk_{i+1} for epoch $i+1$ as well as the update token $\mathsf{up}_{i+1} := \mathsf{cred}_{i+1}$.

$\mathsf{UpdateSK}\ (\mathsf{pp}, \mathsf{sk}_i, \mathsf{up}_{i+1})$ To generate a matching secret for pk_{i+1}, this algorithm takes input the secret key sk_i and the update token up_{i+1}. It calls $\mathsf{ARKG.DeriveSK}$ to compute sk_{i+1} based on the update information.

$\mathsf{Encrypt}\ (\mathsf{pp}, \mathsf{pk}_i, \mathsf{m})$ For any time period, the encryption algorithm takes as input a public key pk_i for the epoch i and a message m. It uses the $\mathsf{KP\text{-}PKE}$ algorithm Encrypt on both of these inputs to compute the ciphertext c.

$\mathsf{Decrypt}\ (\mathsf{pp}, \mathsf{sk}_i, \mathsf{c})$ For any candidate ciphertext c from epoch i, Decrypt takes as input the secret sk_i and uses $\mathsf{KP\text{-}PKE.Decrypt}$ to recover the message m. It returns $\perp$ if decryption fails.

```
Setup(1^λ)
1 : $\mathsf{pp}_0 \leftarrow \mathsf{ARKG.Setup}(\lambda)$
2 : $\mathsf{pp}_1 \leftarrow \mathsf{KP\text{-}PKE.Setup}(\lambda)$
3 : **return** $\mathsf{pp} := (\mathsf{pp}_0, \mathsf{pp}_1)$

Encrypt(pp, pk_i, m)
1 : $\mathsf{c} \leftarrow \mathsf{KP\text{-}PKE.Encrypt}(\mathsf{pk}_i, \mathsf{m})$
2 : **return** $\mathsf{c}$

Decrypt(pp, sk_i, c)
1 : $\mathsf{m} \leftarrow \mathsf{KP\text{-}PKE.Decrypt}(\mathsf{sk}_i, \mathsf{c})$
2 : **return** $\mathsf{m}$

KeyGen(pp)
1 : $(\mathsf{sk}_0, \mathsf{pk}_0) \leftarrow \mathsf{ARKG.KeyGen}(\mathsf{pp})$
2 : **return** $(\mathsf{sk}_0, \mathsf{pk}_0)$

UpdatePK(pp, pk_i)
1 : $\mathsf{pk}_{i+1}, \mathsf{cred}_{i+1} \leftarrow \mathsf{ARKG.DerivePK}(\mathsf{pp}, \mathsf{pk}_i)$
2 : **return** $\mathsf{pk}_{i+1}, \mathsf{cred}_{i+1}$

UpdateSK(pp, sk_i, cred_{i+1})
1 : $\mathsf{sk}_{i+1} \leftarrow \mathsf{ARKG.DeriveSK}(\mathsf{pp}, \mathsf{sk}_i, \mathsf{cred}_{i+1})$
2 : **return** $\mathsf{sk}_{i+1}$
```

Figure 15: Generic construction for Updatable Public-Key Encryption.

Correctness. We note that our construction is correct if the underlying building blocks are correct.

Security Analysis. We now state and prove the main security theorem for UPKE. In the following theorem, it would seem to natural to consider the composability theorem of ARKG, stated in Frymann et al. [FGK+20], to argue security of our protocol. Intuitively, it says that ARKG can be composed with any compatible public key protocol without loss of secrity (assuming PK-unlinkability). However, we stress that this result reduces security of a composed scheme (e.g. ARKG + PKE) to that of PKE—whereas our proof is for UPKE and thus has different security properties. Hence, a new proof is required.

Theorem VII.12. *The* UPKE *construction given in Figure 15 is* IND-CR-CPA *secure if* ARKG *is PK-unlinkable and has Simulatable Key Derivation, and* KP-PKE *is* IND-IK-CPA *secure.*

Proof. Define Game_0 to be the IND-CR-CPA experiment as defined in Figure 14. Then, define Game_1 as Game_0 with the exception that lines 5 and 6 of the experiment are replaced with $(\mathsf{sk}^*, \mathsf{pk}^*) \leftarrow \mathsf{PKE.KeyGen(pp)}$. The PK-unlinkability property of ARKG ensures that this change is undetectable by an adversary. Furthermore, the update token and key pair provided to the adversary in line 7 is independent of the challenge bit b. Thus, we have:

$$|\Pr[\mathsf{Game}_1 = 1] - \Pr[\mathsf{Game}_0 = 1]| \leqslant \mathsf{Adv}^{\mathsf{PKU}}_{\mathscr{B}_1,\mathsf{ARKG}}(1^\lambda).$$

Define Game_2 as Game_1 with the following change, on line 3, it generates a fresh key pk^* and computes the challenge ciphertext as "$\mathsf{c}^* \leftarrow \mathsf{Encrypt}(\mathsf{pk}^*, \mathsf{m}^*)$". If $\mathscr{A}$ can distinguish this change, then we can construct an adversary $\mathscr{B}_2$ against IND-IK-CCA of KP-PKE. Intuitively, $\mathscr{B}_2$ invokes its experiments to receive its two challenge public keys $\mathsf{pk}^{(0)}$ and $\mathsf{pk}^{(1)}$, without loss of generality, it guesses for which time interval $\mathscr{A}$ will stop, say at epoch i, and set $\mathsf{pk}_i := \mathsf{pk}^{(0)}$. We bound the maximum number of update queries to be i. It then computes pk_j for $j \in [1, i]$ as $(\mathsf{pk}_{j-1}, \mathsf{cred}_{j-1}, r_{j-1}) = \mathsf{T\text{-}Derive}(\mathsf{pk}_j)$ (which exists due to the assumption ARKG has simulatable key derivation). It stores these in a list KL by setting $\mathsf{sk}_{j-1} = \perp$ and $\mathsf{up}_{j-1} = (\mathsf{cred}_j, r_{j-1})$. When called, the update oracle $\mathscr{O}_{\mathsf{upd}}$ instead returns these values upon query.

It waits for $\mathscr{A}$ to select two messages $\mathsf{m}^{(0)}$ and $\mathsf{m}^{(1)}$, and forwards them to its own game. It receives back a challenge ciphertext c^* that it forwards to $\mathscr{A}$. It computes the updated keys according to the description of the game, passes these to $\mathscr{A}$ and waits for a response. We observe that if $b = 0$ then the game coincides with Game_0, otherwise $\mathscr{A}$ is playing against Game_1, thus if $\mathscr{A}$ can distinguish the games, then $\mathscr{B}_2$ can win its IND-IK-CCA game. After some time $\mathscr{A}$ outputs a guess at bit b, which $\mathscr{B}$ forwards as a guess to its own game. Since the simulation does not fail unless $\mathscr{B}$ embedded the challenge keys in the wrong epoch, then $\mathscr{B}$ wins whenever $\mathscr{A}$ wins. Thus, the advantage of the distinguisher is bound by the advantage $\mathscr{B}_2$ has against IND-IK-CCA.

$$|\Pr[\mathsf{Game}_2 = 1] - \Pr[\mathsf{Game}_1 = 1]| \leqslant \frac{1}{q_o}\mathsf{Adv}^{\mathsf{IND\text{-}IK\text{-}CCA}}_{\mathscr{B}_2,\mathsf{KP\text{-}PKE}}(1^\lambda)$$

The experiment is now independent of the challenge bit, and therefore from the sequence of games Game_0 to Game_2, we conclude the advantage of $\mathscr{A}$ against IND-CCA of UPKE is negligble.

$$\mathsf{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathscr{A},\mathsf{UPKE}}(1^\lambda) \leqslant \frac{1}{i}\mathsf{Adv}^{\mathsf{IND\text{-}IK\text{-}CCA}}_{\mathscr{B}_2,\mathsf{KP\text{-}PKE}}(1^\lambda) + \mathsf{Adv}^{\mathsf{PKU}}_{\mathscr{B}_1,\mathsf{ARKG}}(1^\lambda).$$

□

```
Setup(λ)
1 : $\mathsf{pp}_0 \leftarrow \mathsf{ARKG.Setup}(\lambda)$
2 : $\mathsf{pp}_1 \leftarrow \mathsf{PKE.Setup}(\lambda)$
3 : return $\mathsf{pp} := (\mathsf{pp}_0, \mathsf{pp}_1)$

Encrypt(pp, pk, m)
1 : $\mathsf{pk}', \mathsf{cred} \leftarrow \mathsf{ARKG.DerivePK}(\mathsf{pp}, \mathsf{pk})$
2 : $\mathsf{c} \leftarrow \mathsf{PKE.Encrypt}(\mathsf{pk}, \mathsf{m})$
3 : return $\mathsf{C} = (\mathsf{c}, \mathsf{cred})$

KeyGen(pp)
1 : $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{ARKG.KeyGen}(1^\lambda)$
2 : return $(\mathsf{sk}, \mathsf{pk})$

Decrypt(pp, sk, C)
1 : Parse $\mathsf{C}$ as $(\mathsf{c}, \mathsf{cred})$
2 : $\mathsf{sk}' \leftarrow \mathsf{ARKG.DeriveSK}(\mathsf{pp}, \mathsf{sk}, \mathsf{cred})$
3 : $\mathsf{m} \leftarrow \mathsf{PKE.Decrypt}(\mathsf{sk}', \mathsf{c})$
4 : return $\mathsf{m}$
```

Figure 16: Generic construction for Key-Private Public-Key Encryption.

VII.4 Generic Construction for Key-Private Public Key Encryption

In this section we give a generic construction for a key-private public key encryption scheme. Whilst it may be the case that some popular encryption schemes are already key-private, such as ElGamal and Cramer-Shoup encryption [BBDP01a], we show that with no additional assumptions from what we have already assumed to construct UPKE, we can convert any PKE scheme into one that is key-private. Our only two building blocks are ARKG and standard PKE.

At a high-level, we use ARKG to derive ephemeral encryption keys which cannot be linked to the long-term key due to the PK-unlinkability property this building block provides. We then reduce ciphertext indistinguishability of KP-PKE to the same property of of the underlying PKE. We stress here that ARKG plays a different role—previously we required its asynchronous nature to compute key updates, whereas here we will use its PK unlinkability property to ensure key privacy of PKE. Much like for the generic construction of UPKE's, we assume that there is a compatibility between ARKG and PKE. We give intuitive description for our scheme and present the formal definition in Figure 16.

Setup (1^λ) Setup takes as input a security parameter and executes the setup algorithms for both underlying building blocks ARKG and KP-PKE. It outputs the result in public parameters pp.

KeyGen (pp) The key generation algorithm for UPKE takes as input pp and runs $\mathsf{ARKG.KeyGen}$. It outputs the a key-pair as $(\mathsf{sk}, \mathsf{pk})$.

Encrypt $(\mathsf{pp}, \mathsf{pk}, \mathsf{m})$ The encryption algorithm takes as input a public key pk a message. First, it derives an ephemeral encryption key as pk'. It uses the this key for the PKE encryption algorithm to compute the ciphertext c. Note it does not output the derived key pk' since this is computable with knowledge of sk.

Decrypt $(\mathsf{pp}, \mathsf{sk}, \mathsf{c})$ On input of a ciphertext c, Decrypt derives the correct ephemeral secret key sk' from $\mathsf{ARKG.DeriveSK}$. It then uses this key in the algorithm $\mathsf{PKE.Decrypt}$ to recover the message m.

Correctness. Correctness of this construction once again follows from the correctness of ARKG and PKE.

Security Analysis. We now state the security theorem for KP-PKE. We prove two statements that capture chosen plaintext attacks and chosen ciphertext attacks. Our protocol meets the stronger $\mathsf{IND\text{-}IK\text{-}CCA}$ definition if and only if PKE is $\mathsf{IND\text{-}CCA}$. Finally we note that both results are in the standard model.

Theorem VII.13. *The construction given in Figure 16 is:*

a) $\mathsf{IND\text{-}IK\text{-}CCA}$ *secure if ARKG is PK-unlinkable and PKE is* $\mathsf{IND\text{-}CCA}$.

b) $\mathsf{IND\text{-}IK\text{-}CPA}$ *secure if ARKG is PK-unlinkable and PKE is* $\mathsf{IND\text{-}CPA}$.

Proof. a) Let Game_0 be defined as the $\mathsf{IND\text{-}IK\text{-}CCA}$ experiment in Figure 13. For Game_1, we change line 4 of the experiment to be "$\mathsf{pk}^* \leftarrow \mathsf{KeyGen},\ \mathsf{c}^* \leftarrow \mathsf{Encrypt}(\mathsf{pk}^*, \mathsf{m})$. That is, we swap the challenge key out for a freshly sampled public key. We argue that the adversary cannot distinguish between these games without breaking the PK-unlinkability property of ARKG. To see this, let $\mathscr{B}_1$ be the adversary against ARKG. It constructs the experiment against $\mathscr{A}$ as described in Game_0, except that it replaces $\mathsf{pk}^{(b)}$ with its challenge key provided by the PK-unlinkability experiment. We have the game coincides with Game_0 if $b = 0$ in the PK-unlinkability game, otherwise it coincides with the same game but for when $b = 1$. Thus, if $\mathscr{A}$ is able to distinguish Game_0 and Game_1 then $\mathscr{B}_1$ can win the PK-unlinkability experiment.

$$|\Pr[\mathsf{Game}_0 = 1] - \Pr[\mathsf{Game}_1 = 1]| \leqslant \mathsf{Adv}^{\mathsf{PKU}}_{\mathscr{B}_1,\mathsf{ARKG}}(1^\lambda)$$

Now that the challenge ciphertext does not depend on the key, only the message, we argue that an adversary that is able to win Game_1 can be used to win an $\mathsf{IND\text{-}CCA}$ experiment against PKE. To see this, define an $\mathsf{IND\text{-}CCA}$ adversary $\mathscr{B}_2$ against PKE. It sets up the game according to Game_1 except for computation of the challenge ciphertext on line 4. Instead, it sets pk^* to be the challenge key from the $\mathsf{IND\text{-}CCA}$ game, and forwards any queries to $\mathscr{O}_{\mathsf{dec}}$ to the decryption oracle provided by the $\mathsf{IND\text{-}CCA}$ game. It obtains the decrypted message m which it can pass to $\mathscr{A}$.

Then $\mathscr{A}$ outputs two messages $\mathsf{m}_0, \mathsf{m}_1$ which $\mathscr{B}_2$ forwards to the $\mathsf{IND\text{-}CCA}$ challenger. It then sets c^* to be the corresponding challenge ciphertext output from the same experiment. Then, $\mathscr{A}$ submits its guess at the challenge bit b, which $\mathscr{B}_2$ forwards as its response to the $\mathsf{IND\text{-}CCA}$ game. It wins if and only if $\mathscr{A}$ wins its $\mathsf{IND\text{-}IK\text{-}CCA}$ experiment.

$$|\Pr[\mathsf{Game}_1 = 1]| \leqslant \mathsf{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathscr{B}_2,\mathsf{PKE}}(1^\lambda)$$

Thus, from the sequence of games, we conclude the advantage is bound by an adversary against the PK-unlinkability property of ARKG and $\mathsf{IND\text{-}CCA}$ property of PKE.

$$\mathsf{Adv}^{\mathsf{IND\text{-}IK\text{-}CCA}}_{\mathscr{A},\mathsf{KP\text{-}PKE}}(1^\lambda) \leqslant \mathsf{Adv}^{\mathsf{PKU}}_{\mathscr{B}_1,\mathsf{ARKG}}(1^\lambda) + \mathsf{Adv}^{\mathsf{IND\text{-}CCA}}_{\mathscr{B}_2,\mathsf{PKE}}(1^\lambda)$$

b) For brevity we do not recall the full proof but note that it follows the same approach as a) but $\mathscr{A}$ does not have access to the $\mathsf{IND\text{-}IK\text{-}CCA}$ decryption oracle. This means the does not need the decryption oracle from its $\mathsf{IND\text{-}CCA}$ experiment either, and thus $\mathsf{IND\text{-}CPA}$ security for PKE is sufficient. □

By combing our results from Sections VII.3 and VII.4, we obtain the following corollary.

Corollary VII.14. *ARKG that has PK-unlinkability and Simulatable Key Derivation, and PKE with* IND-CPA *security implies* IND-CR-CPA*-secure UPKE.*

VII.5 Instantiation with Cryptographic Group Actions

We show how to construct an ARKG scheme from cryptographic group actions as formally defined in §VII.2.1. We show how to do this both in the setting where the group action is effective and restricted effective. The underlying idea is inspired by the original scheme based on discrete logarithms proposed by Frymann et.al. [FGK+20], however we are required to circumvent a multiplicative issue that arises in the setting of group actions. Intuitively, we solve this by replacing this multiplication operation by composition.

VII.5.1 PRF-OGADH Assumption

The PRF-ODH assumption was introduced to analyse the security of TLS 1.3 [BFGJ17], but since has found application in many other protocols including the original ARKG scheme [FGK+20]. We introduce and formalise a new assumption in the context of cryptographic group actions which we call the "PRF oracle-group-action-Diffie-Hellman" (PRF-OGADH) assumption. Our definition closely follows that of the original PRF-ODH assumption but moved to the group action setting.

Definition VII.15 (PRF-OGADH assumption)**.** *Let $(G, X, \star)$ be a cryptographic group action*[9]*. Let $f : X \times \{0,1\}^* \to \{0,1\}^\lambda$ be a pseudorandom function that takes as input an element $x \in X$ (often referred to as the key) and a label $\kappa \in \{0,1\}^*$ and outputs the evaluation $\tau = f(x,\kappa)$. We define a generic security notion* $\mathrm{lrPRF-OGADH}$ *which is parameterised by* $\mathrm{l}, \mathrm{r} \in \{\mathrm{n}, \mathrm{s}, \mathrm{m}\}$ *indicating how often the adversary is allowed to query a certain "left" or "right" oracle. These oracles are called* OGADH_g *and* OGADH_h *respectively, indexed by group elements $g, h \in G$, and are defined as follows:*

- $\mathrm{OGADH}_g(\mathrm{l}, B)$*: If* $\mathrm{l} = \mathrm{n}$ *(*s *and* m *respectively) then no (a single and arbitrarily many respectively) query(ies) can be made to this oracle. On a query of the form (y,κ) if either $y \notin X$ or $(y,\kappa) = B$ then return $\perp$. Otherwise, it computes $\tau := f(g \star y, \kappa)$ and returns τ.*
- $\mathrm{OGADH}_h(\mathrm{r}, B)$*: If* $\mathrm{r} = \mathrm{n}$ *(*s *and* m *respectively) then no (a single and arbitrarily many respectively) query(ies) can be made to this oracle. On a query of the form (y',κ) if either $y' \notin X$ or $(y',\kappa) = B$ then return $\perp$. Otherwise, it computes $\tau := f(h \star y', \kappa)$ and returns τ.*

Consider the following security game PRF-OGADH between a challenger and a PPT adversary $\mathscr{A}$:

1. *The challenger samples a group element $g \in G$, computes the action $y := g \star x_0$ and sends x_0, y to the adversary $\mathscr{A}$;*
2. *If* $\mathrm{l} = \mathrm{m}$*, $\mathscr{A}$ can issue arbitrary many queries to the oracle* $\mathrm{OGADH}_g(\mathrm{l}, \mathrm{null})$*;*
3. *The adversary issues a challenge query $\kappa^* \in \{0,1\}^*$ and sends to the challenger. On receipt, the challenger samples uniformly at random a group element $h \in G$ and a bit $b \in \{0,1\}$.*

[9]Referring to whether you are in the restricted setting or not.

It then computes $\tau_0 = f(h \star y, \kappa^*)$ and samples $\tau_1 \in \{0,1\}^\lambda$ uniformly at random. The challenge sends $(h \star x_0, \tau_b)$ to the adversary $\mathscr{A}$;

4. *The adversary may issue queries to the oracles* $\mathrm{OGADH}_g(\mathrm{l}, (h \star x_0, \kappa^*))$ *and* $\mathrm{OGADH}_h(\mathrm{r}, (g \star x_0, \kappa^*))$*;*
5. *At some point the adversary stops and outputs a guess bit $b' \in \{0,1\}$.*

We say the adversary wins the lrPRF − OGADH *game if $b' = b$ and define the advantage function*

$$\mathsf{Adv}^{\mathrm{lrprf-ogadh}}_{\mathsf{PRF},\mathscr{A}}(\lambda) := |2 \cdot \Pr\left[b' = b\right] - 1|$$

Moreover, we say the cryptographic group action $(G, X, \star)$ achieves lrPRF − OGADH *security if for any adversary $\mathscr{A}$ the advantage defined above is negligible in the security parameter.*

Among the various settings defined in Definition VII.15, for the purposes of our upcoming constructions, we will need the nnPRF − OGADH and snPRF − OGADH assumptions. Understanding how these assumptions relate to other well studied assumptions is important in accessing the hardness of the assumption. We show that the nnPRF − OGADH assumption is hard if both the group action decisional Diffie Hellman assumption as well as the security of the PRF is hard. In addition we show that the mnPRF − OGADH is as hard as the group action strong computational Diffie Hellman assumption in the random oracle. Recall from §VII.2.1 that this asks us to compute $(g \cdot h) \star x$ given the triple $(x, g \star x, h \star x)$ and access to a decisional Diffie Hellman oracle. It therefore follows trivially that the snPRF − OGADH is hard given the hardness of StGADH in the ROM. The reductions closely follow the methodology used by Brendel et al. [BFGJ17] in their corresponding reductions. Moreover, they give a much larger range of reductions which we leave as future work.

We start by discussing the nnPRF − OGADH security.

Proposition VII.16. *Let $(G, X, \star)$ be a cryptographic group action (either effective or restricted). If a function $f : X \times \{0,1\}^* \to \{0,1\}^\lambda$ is a secure* PRF *and the* GA − DDH *assumption holds for the group action $(G, X, \star)$, then f is also* nnPRF − OGADH*-secure. More precisely, for any probabilistic polynomial time adversary $\mathscr{A}$ against the* nnPRF − OGADH *security, there are probabilistic polynomial time algorithms $\mathscr{B}_1$ and $\mathscr{B}_2$ such that*

$$\mathrm{Adv}^{\mathrm{nnPRF-OGADH}}_{f,\mathscr{A}}(\lambda) \leq 2 \cdot \mathrm{Adv}^{\mathrm{GA-DDH}}_{(G,X,\star),\mathscr{B}_1}(\lambda) + 2 \cdot \mathrm{Adv}^{\mathrm{PRF}}_{f,\mathscr{B}_2}(\lambda).$$

Proof. The proof strategy adopted here uses a game-hopping technique.

Let Game1 be the original nnPRF − OGADH game but we replace the key $(g \cdot h) \star x$ used to compute the challenge value y_0 by an independent random set element $\hat{h} \star x$. We claim that an adversary $\mathscr{A}$ against the nnPRF − OGADH security cannot distinguish between the original nnPRF − OGADH game and Game1 efficiently with non-negligible advantage, since otherwise there exists an efficient adversary $\mathscr{B}_1$ that can solve the GA − DDH with non-negligible probability. So assume that $\mathscr{A}$ can distinguish the two games. Then $\mathscr{B}_1$ is constructed as follows: it receives its challenge, say $(x, g \star x, h \star x, y')$, and to decide whether $y' = (g \cdot h) \star x$, $\mathscr{B}_1$ runs $\mathscr{A}$ as a subroutine on input $(x, g \star x)$. Then $\mathscr{B}_1$ answers $\mathscr{A}$'s challenge query κ^* with $(h \star x, y_b)$ where $y_0 = f(y', \kappa^*)$ and y_1 is sampled at random from $\{0,1\}^\lambda$ and b is a random bit chosen by $\mathscr{B}_1$. Eventually $\mathscr{A}$ outputs a bit b', and $\mathscr{B}$ returns 0 if and only if $b = b'$. Hence if $\mathscr{A}$ can efficiently distinguish these games then $\mathscr{B}_1$ can also efficiently solve its GA — DDH challenge. Moreover the nnPRF — OGADH advantage can be bounded[10] as follows:

$$\mathrm{Adv}^{\mathrm{nnPRF-OGADH}}_{f,\mathscr{A}}(\lambda) \leq \mathrm{Adv}^{\mathrm{Game1}}_{f,\mathscr{A}}(\lambda) + 2 \cdot \mathrm{Adv}^{\mathrm{GA-DDH}}_{(G,X,\star),\mathscr{B}_1}(\lambda).$$

Now let Game2 be the previous game, but this time we replace the challenge value y_0 itself by a uniform random value in $\{0,1\}^\lambda$. We show that if there exists an efficient adversary $\mathscr{A}$ that can distinguish Game2 from Game1, then there exists an efficient algorithm $\mathscr{B}_2$ that can break the PRF security of f. To initiate the environment for $\mathscr{A}$, $\mathscr{B}_2$ chooses some arbitrary element $g \star x$ and forwards it to $\mathscr{A}$. At some point $\mathscr{A}$ asks the challenge query κ^*, which $\mathscr{B}_2$ relays to its own challenger, receiving the PRF-challenge $y_{\hat{b}}$. Then $\mathscr{B}_2$ forwards $y_{\hat{b}}$ along with an arbitrarily chosen set element $h \star x$ to $\mathscr{A}$. Eventually, $\mathscr{A}$ stops and outputs a bit b'. Algorithm $\mathscr{B}_2$ outputs the same bit b'. Hence if $\mathscr{A}$ can efficiently distinguish these games then $\mathscr{B}_2$ can also efficiently distinguish between the PRF values. Moreover the Game1 advantage can be bounded as follows:

$$\mathrm{Adv}^{\mathrm{Game1}}_{f,\mathscr{A}}(\lambda) \leq \mathrm{Adv}^{\mathrm{Game2}}_{f,\mathscr{A}}(\lambda) + 2 \cdot \mathrm{Adv}^{\mathrm{PRF}}_{f,\mathscr{B}_2}(\lambda).$$

Since y_0 and y_1 are now drawn independently and at random from $\{0,1\}^\lambda$, we have $\mathrm{Adv}^{\mathrm{Game2}}_{f,\mathscr{A}}(\lambda) = 0$, which proves the result. $\square$

For the next property we require the GA — StCDH assumption which is defined in Section VII.2.1.

Proposition VII.17. *Let $(G, X, \star)$ be a cryptographic group action (either effective or resticted). Working in the random oracle model, if the StGADH assumption holds for the group action then we have* mnPRF — OGADH-*security for a random oracle $f : X \times \{0,1\}^* \to \{0,1\}^\lambda$. More precisely, for any probabilistic polynomial time adversary $\mathscr{A}$ against the* mnPRF — OGADH *security, there is a probabilistic polynomial time algorithm $\mathscr{B}$ such that*

$$\mathrm{Adv}^{\mathrm{mnPRF-OGADH}}_{f,\mathscr{A}}(\lambda) \leq \mathrm{Adv}^{\mathrm{GA-StCDH}}_{(G,X,\star),\mathscr{B}}(\lambda).$$

Proof Sketch. First of all, $\mathscr{B}$ obtains the set elements $x, g \star x, h \star x$ as given in the StGADH experiement. Recall that the goal of the algorithm $\mathscr{B}$ is to compute the element $(g \cdot h) \star x$ given access to some decisional Diffie Hellman oracle. To initiate the mnPRF — ODH game environment, the adversary $\mathscr{B}$ sends $x, g \star x$ to $\mathscr{A}$ as input. $\mathscr{A}$ now has access to the random oracle, f, and the OGADH_g oracle. In other words, $\mathscr{A}$ may send queries of the form (y, κ) . To provide an appropriate simulation it must be ensured that, if $\mathscr{A}$ first queries some (y, κ) to OGADH_g and then $(g \star y, \kappa)$ to the random oracle, the answer of the random oracle is consistent with the simulation of OGADH_g, and vice versa. This can be achieved if $\mathscr{B}$ can program the random oracle and has access to a decisional Diffie Hellman oracle GA — DDH$(g \star x, \cdot, \cdot)$.

Simulation of f. The answers of the random oracle f need to be consistent, i.e., if a query is asked repeatedly, f returns the same answer. This can be ensured by standard bookkeeping techniques. If a previously unseen query (y, κ) is received, $\mathscr{B}$ must consider the case that $\mathscr{A}$

[10]the factor 2 in the GA — DDH advantage accounts for moving from a random choice of either giving $\mathscr{B}_1$ the set element $(g \cdot h) \star x$ or a random set element y'.

has already queried $(g \star y, \kappa)$ to OGADH_g. Thus, when receiving a call (y, κ) to f, $\mathscr{B}$ queries its $\mathrm{GA-DDH}$ oracle with $(g \star x, y, \hat{y})$ as input for any set element $\hat{y} \in X$ that has been queried with κ to OGADH_g. If the $\mathrm{GA-DDH}$ oracle returns 1 on any such input then $\mathscr{B}$ answers consistently with the corresponding answer from earlier and otherwise $\mathscr{B}$ assigns a fresh value y_0 to $(\hat{y}, \kappa)$ and returns y_0 to $\mathscr{A}$.

Simulation of OGADH_g**.** Analogously to the simulation of the random oracle, $\mathscr{B}$ checks each newly received request by $\mathscr{A}$ against all previous query-response pairs of OGADH_g and answers consistently in case of repetition. If a previously unseen query (y, κ) is received by OGADH_g, $\mathscr{B}$ must further check whether the related value $(g \star y, \kappa)$ has been queried to the random oracle before. Similar to the reverse case, $\mathscr{B}$ uses its $\mathrm{GA-DDH}$ oracle on $(y, \hat{y})$ on all previous random oracle queries $(\hat{y}, \kappa)$ to detect this. If $\mathrm{GA-DDH}(g \star x, y, \hat{y}) = 1$ for some K, the simulation of OGADH_g answers with the respective output of the random oracle. Otherwise, a response y_1 is drawn uniformly at random from $\{0,1\}^\lambda$ and returned to $\mathscr{A}$.

Completion of the reduction. At some point, $\mathscr{A}$ issues a challenge query κ^* to its challenger. $\mathscr{B}$ answers this query with $h \star x$ and some value y_1, drawn at random from $\{0,1\}^\lambda$. Adversary $\mathscr{A}$ can now query OGADH_g and the random oracle f further, with the limitation that it may not query the pair $(h \star x, \kappa^*)$ to OGADH_g. These queries are simulated as before. Eventually $\mathscr{A}$ stops and outputs a guess bit b'. Then $\mathscr{B}$ queries $\mathrm{GA-DDH}(g \star x, h \star x, \hat{y})$ for all queries $(\hat{y}, \kappa^*)$ of $\mathscr{A}$ to the random oracle. If $\mathrm{GA-DDH}(g \star x, h \star x, \hat{y}) = 1$ for some $(\hat{y}, \kappa^*)$ then $\mathscr{B}$ outputs $\hat{y}$ in the StGADH experiement. Therefore, if the adversary $\mathscr{A}$ wins the $\mathrm{mnPRF-OGADH}$ game with non-neglibile probability, then $\mathscr{B}$ also outputs the correct value $(g \cdot h) \star x$ with non-negligible probability. $\mathscr{B}$ is efficient, since $\mathscr{A}$ is efficient and asks at most polynomially many (with respect to the security parameter) queries to each oracle. □

Remark VII.18. *With minor changes to the details of this reduction, it is straight forward to show that in the random oracle model one gets* $\mathrm{nmPRF-OGADH}$*-security if the* StGADH *assumption holds.*

VII.5.2 An ARKG Construction from Effective Group Actions

We first focus on the construction of ARKG from effective cryptographic group actions. That is, group actions where there is an efficient algorithm for computing the output of the group action, $g \star x$, for any $g \in G$ and $x \in X$.

Our core idea is to append another group element $h \in G$ onto the long term public key $g \star x$. This gives a new derived public key as $h \star (g \star x)$. By the defining property of a group action, its corresponding derived secret key would be the group element hg. We formally detail the algorithms for this construction are specified in Figure 17.

Note that we include the use of a MAC, as in the original discrete-log scheme. This was to facilitate identification of corresponding credentials cred. We have included it in our algorithms for completeness, but note that it is not required in our use for updatable public key encryption.

```
Setup(1^λ)
  return pp := ((G, X, ⋆, x_0 ∈ X),
    MAC, KDF_1, KDF_2)
```

```
KeyGen(pp)
  1: $g \leftarrow\$ G$
  2: $y \leftarrow g \star x_0$
  3: return (sk, pk) := $(g, y)$
```

```
Check(pp, sk'pk')
  Parse sk' = g, pk' = y
  return $y \equiv g \star x_0$
```

```
DerivePK(pp, pk = y)
  1: $(g', y') \leftarrow$ KeyGen(pp)
  2: $z \leftarrow g' \star y$
  3: $K_{mac} \leftarrow \mathsf{KDF}_1(z)$
  4: $K_{cred} \leftarrow \mathsf{KDF}_2(z)$
  5: $\mu \leftarrow \mathsf{MAC}(K_{mac}; y')$
  6: $h \leftarrow \mathsf{Encode}(K_{cred}) \in G$
  7: return (pk' := $h \star y$), $y'$, $\mu$
```

```
DeriveSK(pp, sk = g, y', μ*))
  1: $z \leftarrow g \star y'$
  2: $K_{mac} \leftarrow \mathsf{KDF}_1(z)$
  3: $K_{cred} \leftarrow \mathsf{KDF}_2(z)$
  4: if $\mu^\star \equiv \mathsf{MAC}(K_{mac}; y')$ then
  5:     $h \leftarrow \mathsf{Encode}(K_{cred}) \in G$
  6:     return sk' := $gh$
  7: else return $\perp$
```

Figure 17: ARKG construction from an effective group action with an abelian group G. $\mathsf{Encode}(\cdot)$ is a deterministic function or algorithm which encodes the input as an element in G.

Correctness. Note that the correctness of the scheme is guaranteed since G is abelian: for any $g, g' \in G$ and any $x \in X$ we have $g \star (g' \star x) = g' \star (g \star x)$.

Security Analysis. We show that the described construction satisfies the unlinkability property by reducing it down to the PRF-OGADH assumption that was introduced in §VII.5.1. For completeness, we also show that the secrecy property by reducing it down to the traditional GAIP assumption defined in §VII.2.1

Theorem VII.19 (PK-unlinkability). *For an effective group action $(G, X, \star)$, if the* nnPRF − OGADH *and* GAIP *assumptions hold on the group action, then the ARKG scheme described in Figure 17 satisfies the PK-unlinkability property.*

Proof. $\mathcal{G}_0$ is defined exactly by $\mathsf{Exp}^{\mathsf{PKU}}_{\mathsf{ARKG},\mathcal{A}}(\lambda)$. Thus

$$\Pr\left[\mathcal{G}_0^b = 1\right] = \Pr\left[\mathsf{Exp}^{\mathsf{PKU}}_{\mathcal{A}}(\lambda) = 1\right]$$

We immediately begin by combining, when the challenge bit b is set to 0, the secret key derivation performed by the DeriveSK algorithm with DerivePK performed by the oracle. This is a semantic change and hence there is no loss of advantage to the adversary as the outputs of the oracle are indistinguishable. We now define a series of hybrid games $\mathcal{H}_i$ such that $\mathcal{H}_0 := \mathcal{G}_0$ and $\mathcal{H}_i$ as $\mathcal{H}_{i-1}$ but with the exception that, in the ith oracle call to $\mathcal{O}^b_{\mathsf{pk}'}$, computation of pk′ is replaced with '$\hat{g} \leftarrow\$ G$, $\mathsf{pk}' \leftarrow h \star (\hat{g} \star x_0)$'. Note that $\hat{g}h$ is now returned as sk′. The adversary is unable to distinguish between $\mathcal{H}_i$ and $\mathcal{H}_{i-1}$ as the random sample of h in $\mathcal{H}_{i-1}$ ensures the distribution of $(\mathsf{sk}'_i, \mathsf{pk}'_i)$ is uniformly random in both games, giving

$$|\Pr[\mathcal{H}_k = 1]| = |\Pr[\mathcal{H}_{k-1} = 1]|$$

Next, we continue by defining another series of hybrid games $\tilde{\mathcal{H}}_j$ where $\tilde{\mathcal{H}}_0 := \mathcal{G}_1$. Game $\tilde{\mathcal{H}}_j$ is defined to be game $\tilde{\mathcal{H}}_{j-1}$, with the exception that the execution of DerivePK in the jth oracle call is altered when the challenge bit is set to 0. We replace '$\mu \leftarrow \mathsf{MAC}(mk_j, (E, \mathsf{aux}))$' with '$\mu \leftarrow \mathsf{MAC}(mk'_j, (E, \mathsf{aux}))$', where mk'_j is a uniformly sampled MAC key independent from mk_j. The adversary is able to distinguish between the games $\tilde{\mathscr{H}}_j$ and $\tilde{\mathscr{H}}_{j-1}$ if it is able to forge the MAC key mk_j, where mk_j is the MAC key from $\tilde{\mathscr{H}}_j$. We show that the adversary's ability to do this is bounded by the nnPRF-OGADH property of KDF_2.

The adversary $\mathscr{B}$, against the nnPRF-OGADH game, plays the role of challenger in $\tilde{\mathscr{H}}_j$ for $\mathscr{A}$. It invokes its own game, receiving $(G, X, \star, x_0 \in X), y \leftarrow g \star x_0$ for a uniformly random $g \in G$, and the nnPRF-OGADH challenge $(\tilde{y} \leftarrow g' \star x_0, \tau^\star)$, and sets the challenge label κ as the label for KDF_2. It wins its own game if it can decide whether $b = 0$ or $b = 1$. $\mathscr{B}$ invokes $\tilde{\mathscr{H}}_j$ and sets $\mathsf{pk}_0 \leftarrow y, \mathsf{sk}_0 \leftarrow \perp$. It answers the jth oracle query to $\mathscr{O}^b_{\mathsf{pk}'}$ honestly except it sets $\mathsf{pk}_1 = \tilde{y}$, $\mathsf{sk}_1 = \perp$, and $mk_j \leftarrow \tau_{b,j}$, which is the output of KDF_2 in game $\tilde{\mathscr{H}}_j$. It then waits for $\mathscr{A}$ to output a bit b. It forwards b as the answer to its own nnPRF-OGADH game and wins with probability equal to that of $\mathscr{A}$ distinguishing $\tilde{\mathscr{H}}_j$ from $\tilde{\mathscr{H}}_{j-1}$. Thus, we have

$$\left|\Pr\left[\tilde{\mathscr{H}}_j = 1\right] - \Pr\left[\tilde{\mathscr{H}}_{j-1} = 1\right]\right| \leqslant \mathsf{Adv}^{\mathsf{nnPRF-OGADH}}_{\mathsf{KDF}^i_2, \mathscr{B}}(\lambda)$$

We define $\mathscr{G}^b_2 := \tilde{\mathscr{H}}_{q_o}$, where q_o is the number of queries made to the $\mathscr{O}^b_{\mathsf{pk}'}$ oracle. Next, we define a third series of hybrid games $\hat{\mathscr{H}}_k$ where $\hat{\mathscr{H}}_0 := \mathscr{G}^b_2$. Game $\hat{\mathscr{H}}_k$ is defined to be game $\hat{\mathscr{H}}_{k-1}$, with the exception that the execution of $\mathsf{DerivePK}$ in the k^{th} oracle call is altered. We replace '$h \leftarrow \mathsf{Encode}(\mathsf{KDF}_1(g' \star y))$' with '$h \leftarrow\$ G$'. The advantage of the adversary distinguishing $\hat{\mathscr{H}}_k$ from $\hat{\mathscr{H}}_{k-1}$ is also bound by the nnPRF-OGADH in an argument almost identical to that of $\tilde{\mathscr{H}}_j$ and $\tilde{\mathscr{H}}_{j-1}$. It is omitted here for brevity. Thus we have

$$\left|\Pr\left[\hat{\mathscr{H}}_k = 1\right] - \Pr\left[\hat{\mathscr{H}}_{k-1} = 1\right]\right| \leqslant \mathsf{Adv}^{\mathsf{nnPRF-OGADH}}_{\mathsf{KDF}^i_1, \mathscr{B}}(\lambda)$$

We define $\mathscr{G}^b_3 := \hat{\mathscr{H}}_{q_o}$ and analyse the advantage of $\mathscr{A}$ in distinguishing between $b = 0$ and $b = 1$. Since the game is independent of b, and the distributions of $(\mathsf{sk}', \mathsf{pk}')$ are identical, it follows that

$$\left|\Pr\left[\mathscr{G}^b_3 = 1\right]\right| = \frac{1}{2}$$

Thus the advantage of $\mathscr{A}$ is bounded by

$$\mathsf{Adv}^{\mathsf{PKU}}_{\mathsf{ARKG}, \mathscr{A}}(\lambda) \leqslant q_o \cdot \left(\mathsf{Adv}^{\mathsf{nnPRF-OGADH}}_{\mathsf{KDF}_1, \mathscr{B}}(\lambda) + \mathsf{Adv}^{\mathsf{nnPRF-OGADH}}_{\mathsf{KDF}_2, \mathscr{B}}(\lambda)\right)$$

By assumption that KDF_1 and KDF_2 are nnPRF-OGADH secure, the advantage of their adversaries is negligible and hence the advantage of the adversary against PK-unlinkability is also negligible. □

Theorem VII.20 (SK-secrecy). *Let $(G, X, \star)$ be an effective group action.*

1. *If the* snPRF − OGADH *assumptions hold on the group action then the ARKG construction is msKS and mwKS-secure.*

2. *If the and* GAIP *assumption hold on the group action then the ARKG construction is hsKS and hwKS-secure.*

Proof of Theorem 4.1. $\mathscr{G}_0$ is defined exactly by the experiment $\mathsf{Exp}^{\mathsf{msKS}}_{\mathsf{ARKG},\mathscr{A}}(\lambda)$. Thus

$$\Pr[\mathscr{G}_0 = 1] = \Pr\Big[\mathsf{Exp}^{\mathsf{msKS}}_{\mathsf{ARKG},\mathscr{A}}(\lambda) = 1\Big]$$

Define $\mathscr{G}_1$ as $\mathscr{G}_0$ with the exception that we replace the computation of '$\mathsf{pk}' = h \star y$' in DerivePK with '$\hat{g} \leftarrow\!\$\ G$, $\mathsf{pk}' \leftarrow h \star \hat{g}$' during the oracle call $\mathscr{O}_{\mathsf{pk}'}$. The adversary $\mathscr{B}$ keeps an internal list List that contains elements of the form $(y', g', \hat{g})$. If $\mathscr{A}$ queries $\mathscr{O}_{\mathsf{sk}'}$ with cred $\ni y'$ such that $y' \in$ List then it replaces of DeriveSK run by $\mathscr{O}_{\mathsf{sk}'}$ with '**return** $\mathsf{sk}' = g\hat{g}$' where $\hat{g}$ is obtained from the matching y' entry in List. This ensures that sk' output from the oracle still passes Check when called on a corresponding public key obtained from $\mathscr{O}_{\mathsf{pk}'}$. As both sk' and h are uniformly sampled from the same space, the two games are indistinguishable. Hence

$$\Pr[\mathscr{G}_1 = 1] = \Pr[\mathscr{G}_0 = 1]$$

We then construct an adversary $\mathscr{B}$ for the snPRF-OGADH game from an adversary that is able to win at $\mathscr{G}_1$. That is, break msKS security of ARKG.

The adversary $\mathscr{B}$ gets the challenge $\tau^\star$ from its snPRF-OGADH game. It sets up the game as described except it sets $\mathsf{sk} \leftarrow \perp$, $\mathsf{pk} \leftarrow y$, and the label of KDF_1 to be the challenge label κ from its own game. It challenges $\mathscr{A}$ to create a forgery on y and is able to answer oracle queries honestly. Then, $\mathscr{A}$ outputs the tuple $(\mathsf{sk}^\star, \mathsf{pk}^\star, \mathsf{cred}^\star)$, from which $\mathscr{B}$ can create a successful answer to the snPRF-OGADH challenge $(x_0, y, y', \tau^\star)$. It extracts y' from $\mathsf{cred}^\star$ and uses the single *OGADH* oracle query in the snPRF-OGADH game to get $\tau^\star \leftarrow \mathsf{KDF}_1(g' \star y)$. $\mathscr{B}$ can then compute the secret key as $g = \mathsf{sk} \cdot h^{-1}$. With knowledge of g it is trivial for $\mathscr{B}$ to compute $\tau = \mathsf{KDF}_1(g \star y')$ and make the comparison $\tau \overset{?}{=} \tau^\star$. If they are equal, then $b = 0$, otherwise $b = 1$. The event that $\mathscr{A}$ queries the snPRF-OGADH challenge y' (which would cause the experiment to abort) happens with probability q_o/p where p is the size of G and q_o is the number of oracle queries made by $\mathscr{A}$. This probability is negligible as p is large.

Thus, the advantage of $\mathscr{A}$ in $\mathsf{Exp}^{\mathsf{msKS}}_{\mathsf{ARKG},\mathscr{A}}(\lambda)$ is bound by an adversary $\mathscr{B}$ against the snPRF-OGADH assumption, giving

$$\mathsf{Adv}^{\mathsf{msKS}}_{\mathsf{ARKG},\mathscr{A}}(\lambda) \leqslant \frac{p - q_o}{p} \cdot \mathsf{Adv}^{\mathsf{snprf-ogadh}}_{G,\mathscr{B}}(\lambda)$$

If the snPRF-OGADH assumption is hard in (G, X) then is msKS secure. □

Proof of Theorem 4.2. $\mathscr{G}_0$ is defined exactly by the experiment $\mathsf{Exp}^{\mathsf{hsKS}}_{\mathsf{ARKG},\mathscr{A}}(\lambda)$. Thus

$$\Pr[\mathscr{G}_0 = 1] = \Pr\Big[\mathsf{Exp}^{\mathsf{hsKS}}_{\mathsf{ARKG},\mathscr{A}}(\lambda) = 1\Big]$$

Define $\mathscr{G}_1$ as $\mathscr{G}_0$ with the exception that we replace the computation of '$\mathsf{pk}' = h \star y$' in DerivePK with '$\hat{g} \leftarrow\!\$\ G$, $\mathsf{pk}' \leftarrow \hat{g} \star y$' during the oracle call $\mathscr{O}_{\mathsf{pk}'}$. The adversary $\mathscr{B}$ keeps an internal list List that contains elements of the form $(y', g', \hat{g})$. If $\mathscr{A}$ queries $\mathscr{O}_{\mathsf{sk}'}$ with cred $\ni y'$ such that $y' \in$ List then it replaces of DeriveSK run by $\mathscr{O}_{\mathsf{sk}'}$ with '**return** $\mathsf{sk}' = g\hat{g}$' where $\hat{g}$ is obtained from the matching y' entry in List. This ensures that sk' output from the oracle still passes Check when called on a corresponding public key obtained from $\mathscr{O}_{\mathsf{pk}'}$. As both sk' and h are uniformly sampled from the same space, the two games are indistinguishable. Hence

$$\Pr[\mathscr{G}_1 = 1] = \Pr[\mathscr{G}_0 = 1]$$

We then construct an adversary $\mathcal{B}$ for the GAIP game from an adversary that is able to win at $\mathcal{G}_1$, that is, break hsKS security of ARKG. The adversary $\mathcal{B}$ sets up the game as described in $\mathcal{G}_1$, except that on line 2 it replaces '$(\mathsf{sk},\mathsf{pk}) \leftarrow \mathsf{KeyGen}$' with '$\mathsf{pk} \leftarrow y, \mathsf{sk} \leftarrow \perp$', where y is $\mathcal{B}$'s challenge from the GAIP game. The challenger $\mathcal{B}$ chooses one query to the oracle where it guesses $\mathcal{A}$ will use the derived pk' as part of its forgery. For this query, it can answer oracles calls to $\mathcal{O}_{\mathsf{pk}'}$ using the GAIP challenge y, and cannot answer $\mathcal{O}_{\mathsf{sk}'}$ queries. In the event this oracle is queried, the experiment aborts. For all other queries, $\mathcal{B}$ can answer oracle queries made by $\mathcal{A}$ as $\mathcal{B}$ generates the ephemeral key pair (g', y') and can extract the value $\hat{g}$ from List.

Then, $\mathcal{B}$ waits for $\mathcal{A}$ to output a successful forgery sk' with credential $\mathsf{cred}^\star$. Using y' from $\mathsf{cred}^\star$, $\mathcal{B}$ is able to locate y' in List and find the corresponding g'. This allows $\mathcal{B}$ to recompute ck from $\mathsf{KDF}_1(g' \star y)$ and compute an s such that $s \star x_o = y$ as $s = \mathsf{sk}' \cdot h^{-1}$. $\mathcal{B}$ is guaranteed that $y' \in \mathsf{List}$ as a successful forgery in the hsKS game requires $(\mathsf{pk}^\star, \mathsf{cred}^\star) \in \mathsf{PKList}$ which can only happen if $\mathcal{B}$ generated y' during an oracle call from $\mathcal{A}$. However, the simulation fails if $\mathcal{A}$ queries $\mathcal{O}_{\mathsf{sk}'}$ on pk' that embeds the GAIP challenge, which happens with probability $1/q_o$. Thus, the advantage of $\mathcal{A}$ in $\mathsf{Exp}^{\mathsf{hsKS}}_{,\mathcal{A}}(\lambda)$ is bound by an adversary $\mathcal{B}$ against the Group Action Inverse Problem assumption, giving

$$\mathsf{Adv}^{\mathsf{hwKS}}_{\mathsf{ARKG},\mathcal{A}}(\lambda) \leqslant \frac{1}{q_o}\mathsf{Adv}^{\mathsf{gaip}}_{G,\mathcal{B}}(\lambda)$$

By assumption, the GAIP problem is hard and thus is hsKS secure.

□

Simulatable Key Derivation. Much like the construction of Frymann et al. [FGK+20], we inherently get the simulatable public key derivation property since the underlying group action is free and transitive. This ensures that for every element $y \in X$, there is a unique group element $g \in G$ such that $y = g \star x$. Therefore by applying the action of a random inverse element, $\hat{g}^{-1}$ to a derived public key, the result obtained forms a valid public key.

VII.5.3 An ARKG Construction from Restricted Effective Group Actions

Now we look at the setting when we have a restricted effective group action. Namely, there is a (not necessarily minimal) generating set $S := \{g_i\}_{i=1}^n$ for the group G and an efficient algorithm for computing the output of the group action on g_i and its inverse, $g_i \star x$ and $g_i^{-1} \star x$, for any g_i in S and $x \in X$. Recall that any element of G can be represented as a finite sequence in $\{g_1, \cdots, g_n, g_1^{-1}, \cdots, g_n^{-1}\}^*$, or equivalently, $g \in G$ can be represented as $g = \prod g_i^{a_i}$ for $a_i \in \mathbb{Z}$.

The challenge is that we can not apply the group action to the element g itself. Instead, we recursively apply the group element g_i (or g_i^{-1}) a total of $|a_i|$ times, which will result in the computation of $g \star x$. This also means that secret key space in key generation algorithm versus secret key space in the DeriveSK algorithm will be different. The algorithms for this construction are specified in Figure 18. In the description of these algorithms, we use the notation $C_k = \{-k, \cdots, k\}$ to represent the integers whose absolute value are at most k. In addition, we abuse the notation a few times in how we do the action computations. The key generation algorithm lays out this recursive description in how these group action computations are actually done. In other places we write it as just one group action computation when we actually mean this recursive description.

```
Setup(1^λ)
return pp := ((G = ⟨g_1, ···, g_n⟩, X, ⋆, x_0), k = ⟨k_1, ···, k_n⟩ ∈ $\mathbb{Z}^n_{>0}$
    MAC, KDF_1, KDF_2)
```

```
KeyGen(pp)
1 : $y \leftarrow x_0$
2 : for $i \in \{1, \cdots, n\}$
3 :     $a_i \leftarrow\$ C_{k_i}$
4 :     $y \leftarrow g_i^{a_i} \star y$
5 : $\mathbf{a} \leftarrow \{a_1, \cdots, a_n\}$
6 : return $(\mathsf{sk}, \mathsf{pk}) := (\mathbf{a}, y)$
```

```
Check(pp, sk', pk')
Parse $\mathsf{sk}' = \mathbf{a}, \mathsf{pk}' = y$
return $y \equiv \left(\prod_{i\in\{1,\cdots,n\}} g_i^{a_i}\right) \star x_0$
```

```
DerivePK(pp, pk = y)
1 : $(\mathbf{a}', y') \leftarrow \mathsf{KeyGen}(\mathsf{pp})$
2 : $g' \leftarrow \prod_{i\in\{1,\cdots,n\}} g_i^{a'_i}$
3 : $z \leftarrow g' \star y$
4 : $K_{mac} \leftarrow \mathsf{KDF}_1(z)$
5 : $K_{cred} \leftarrow \mathsf{KDF}_2(z)$
6 : $\mu \leftarrow \mathsf{MAC}(K_{mac}; y')$
7 : $\mathbf{b} \leftarrow \mathsf{Encode}(K_{cred}) \in C_{k_1} \times \cdots \times C_{k_n}$
8 : $h \leftarrow \prod_{i\in\{1,\cdots,n\}} g_i^{b_i}$
9 : return $(\mathsf{pk}' := h \star y), y', \mu$
```

```
DeriveSK(pp, sk = a, y', μ*))
1 : $g \leftarrow \prod_{i\in\{1,\cdots,n\}} g_i^{a_i}$
2 : $z \leftarrow g \star y'$
3 : $K_{mac} \leftarrow \mathsf{KDF}_1(z)$
4 : $K_{cred} \leftarrow \mathsf{KDF}_2(z)$
5 : if $\mu^\star \equiv \mathsf{MAC}(K_{mac}; y')$ then
6 :     $\mathbf{b} \leftarrow \mathsf{Encode}(K_{cred}) \in C_{k_1} \times \cdots \times C_{k_n}$
7 :     return $\mathsf{sk}' := \mathbf{a} + \mathbf{b} = \{a_i + b_i\}$
8 : else return $\perp$
```

Figure 18: ARKG construction from a restricted effective group action with an abelian group G. Encode($\cdot$) *is a deterministic function or algorithm which encodes the input as an element in $C_{k_1} \times \cdots \times C_{k_n}$.*

Security Analysis. We now present the security theorems and proofs for our ARKG construction from restricted group actions.

Theorem VII.21 (PK-unlinkability). *Let $(G, X, \star)$ be a restricted effective group action. If the* nnPRF — OGADH *and* GAIP *assumptions hold on the group action, then the ARKG scheme described in Figure 17 satisfies the PK-unlinkability property.*

Proof. $\mathcal{G}_0$ is defined exactly by $\mathsf{Exp}^{\mathsf{PKU}}_{\mathsf{ARKG},\mathcal{A}}(\lambda)$. Thus

$$\Pr\left[\mathcal{G}_0^b = 1\right] = \Pr\left[\mathsf{Exp}^{\mathsf{PKU}}_{\mathcal{A}}(\lambda) = 1\right]$$

We immediately begin by combining, when the challenge bit b is set to 0, the secret key derivation performed by the DeriveSK algorithm with DerivePK performed by the oracle. This is a semantic change and hence there is no loss of advantage to the adversary as the outputs of the oracle are indistinguishable. We now define a series of hybrid games $\mathcal{H}_i$ such that $\mathcal{H}_0 := \mathcal{G}_0$ and $\mathcal{H}_i$ as $\mathcal{H}_{i-1}$ but with the exception that, in the ith oracle call to $\mathcal{O}^b_{\mathsf{pk}'}$, computation of pk' is replaced with '$\hat{\mathbf{a}} \leftarrow\$ C_{k_1} \times ... \times C_{k_n}, \hat{g} \leftarrow \prod_{i=1}^n g_i^{a_i}, \mathsf{pk}' \leftarrow h \star (\hat{g} \star x_0)$'. Note that $\hat{\mathbf{a}} + \mathbf{b}$ is now returned as sk'. The adversary is unable to distinguish between $\mathcal{H}_i$ and $\mathcal{H}_{i-1}$ as the distribution of h in $\mathcal{H}_{i-1}$ is negligibly close to uniform, by definition of a restricted effective group action. Thus, the distribution of $(\mathsf{sk}'_i, \mathsf{pk}'_i)$ is also uniformly random in both games, giving

$$|\Pr[\mathcal{H}_k = 1]| = |\Pr[\mathcal{H}_{k-1} = 1]|$$

Next, we continue by defining another series of hybrid games $\tilde{\mathscr{H}}_j$ where $\tilde{\mathscr{H}}_0 := \mathscr{G}_1$. Game $\tilde{\mathscr{H}}_j$ is defined to be game $\tilde{\mathscr{H}}_{j-1}$, with the exception that the execution of DerivePK in the jth oracle call is altered when the challenge bit is set to 0. We replace '$\mu \leftarrow \mathsf{MAC}(mk_j, (E, \mathsf{aux}))$' with '$\mu \leftarrow \mathsf{MAC}(mk_j', (E, \mathsf{aux}))$', where mk_j' is a uniformly sampled MAC key independent from mk_j. The adversary is able to distinguish between the games $\tilde{\mathscr{H}}_j$ and $\tilde{\mathscr{H}}_{j-1}$ if it is able to forge the MAC key mk_j, where mk_j is the MAC key from $\tilde{\mathscr{H}}_j$. We show that the adversary's ability to do this is bounded by the nnPRF-OGADH property of KDF_2.

The adversary $\mathscr{B}$, against the nnPRF-OGADH game, plays the role of challenger in $\tilde{\mathscr{H}}_j$ for $\mathscr{A}$. It invokes its own game, receiving $(G, X, \star, x_0), y \leftarrow g \star x_0$, where g is computed as $g \leftarrow \prod_{i=1}^{n} g_i^{r_i}$ for $r_i \leftarrow \$C_{k_i}$. The nnPRF-OGADH challenge is $(\tilde{y} \leftarrow g' \star x_0, \tau^\star)$, and sets the challenge label κ as the label for KDF_2. It wins its own game if it can decide whether $b = 0$ or $b = 1$. $\mathscr{B}$ invokes $\tilde{\mathscr{H}}_j$ and sets $\mathsf{pk}_0 \leftarrow y, \mathsf{sk}_0 \leftarrow \perp$. It answers the jth oracle query to $\mathscr{O}^b_{\mathsf{pk}'}$ honestly except it sets $\mathsf{pk}_1 = y'$, $\mathsf{sk}_1 = \perp$, and $mk_j \leftarrow \tau_{b,j}$, which is the output of KDF_2 in game $\tilde{\mathscr{H}}_j$. It then waits for $\mathscr{A}$ to output a bit b. It forwards b as the answer to its own nnPRF-OGADH game and wins with probability equal to that of $\mathscr{A}$ distinguishing $\tilde{\mathscr{H}}_j$ from $\tilde{\mathscr{H}}_{j-1}$. Thus, we have

$$\left|\Pr\left[\tilde{\mathscr{H}}_j = 1\right] - \Pr\left[\tilde{\mathscr{H}}_{j-1} = 1\right]\right| \leqslant \mathsf{Adv}^{\mathsf{nnPRF-OGADH}}_{\mathsf{KDF}_2^i, \mathscr{B}}(\lambda)$$

We define $\mathscr{G}_2^b := \tilde{\mathscr{H}}_{q_o}$, where q_o is the number of queries made to the $\mathscr{O}^b_{\mathsf{pk}'}$ oracle. Next, we define a third series of hybrid games $\hat{\mathscr{H}}_k$ where $\hat{\mathscr{H}}_0 := \mathscr{G}_2^b$. Game $\hat{\mathscr{H}}_k$ is defined to be game $\hat{\mathscr{H}}_{k-1}$, with the exception that the execution of DerivePK in the k^{th} oracle call is altered. We replace '$h \leftarrow \mathsf{Encode}(\mathsf{KDF}_1(g' \star y))$' with '$h = \prod_{i=1}^{n} g_i^{r_i'}$ for $r_i' \leftarrow \$C_{k_i}$'. The advantage of the adversary distinguishing $\hat{\mathscr{H}}_k$ from $\hat{\mathscr{H}}_{k-1}$ is also bound by the nnPRF-OGADH in an argument almost identical to that of $\tilde{\mathscr{H}}_j$ and $\hat{\mathscr{H}}_{j-1}$. It is omitted here for brevity. Thus we have

$$\left|\Pr\left[\hat{\mathscr{H}}_k = 1\right] - \Pr\left[\hat{\mathscr{H}}_{k-1} = 1\right]\right| \leqslant \mathsf{Adv}^{\mathsf{nnPRF-OGADH}}_{\mathsf{KDF}_1^i, \mathscr{B}}(\lambda)$$

We define $\mathscr{G}_3^b := \hat{\mathscr{H}}_{q_o}$ and analyse the advantage of $\mathscr{A}$ in distinguishing between $b = 0$ and $b = 1$. Since the game is independent of b, and the distributions of $(\mathsf{sk}', \mathsf{pk}')$ are identical, it follows that

$$\left|\Pr\left[\mathscr{G}_3^b = 1\right]\right| = \frac{1}{2}$$

Thus the advantage of $\mathscr{A}$ is bounded by

$$\mathsf{Adv}^{\mathsf{PKU}}_{\mathsf{ARKG}, \mathscr{A}}(\lambda) \leqslant q_o \cdot \left(\mathsf{Adv}^{\mathsf{nnPRF-OGADH}}_{\mathsf{KDF}_1, \mathscr{B}}(\lambda) + \mathsf{Adv}^{\mathsf{nnPRF-OGADH}}_{\mathsf{KDF}_2, \mathscr{B}}(\lambda)\right)$$

By assumption that KDF_1 and KDF_2 are nnPRF-OGADH secure, the advantage of their adversaries is negligible and hence the advantage of the adversary against PK-unlinkability is also negligible. □

Theorem VII.22 (SK-secrecy). *Let $(G, X, \star)$ be a restricted effective group action.*

1. *If the* snPRF − OGADH *assumptions hold on the group action then the ARKG construction is msKS and mwKS-secure.*

2. *If the* GAIP *assumption hold on the group action then the ARKG construction is hsKS and hwKS-secure.*

Proof. $\mathcal{G}_0$ is defined exactly by the experiment $\mathsf{Exp}^{\mathsf{msKS}}_{\mathsf{ARKG},\mathcal{A}}(\lambda)$. Thus

$$\Pr[\mathcal{G}_0 = 1] = \Pr\Big[\mathsf{Exp}^{\mathsf{msKS}}_{\mathsf{ARKG},\mathcal{A}}(\lambda) = 1\Big]$$

Define $\mathcal{G}_1$ as $\mathcal{G}_0$ with the exception that the computation of pk' is replaced with '$\mathbf{\hat{a}} \leftarrow\!\!\$\; C_{k_1} \times \ldots \times C_{k_n}, \hat{g} \leftarrow \prod_{i=1}^{n} g_i^{a_i}, \mathsf{pk}' \leftarrow h \star (\hat{g} \star x_0)$'. The adversary $\mathcal{B}$ keeps an internal list List that contains elements of the form $(y', g', \hat{g})$. If $\mathcal{A}$ queries $\mathcal{O}_{\mathsf{sk}'}$ with cred $\ni y$ such that $y \in$ List then it replaces sk' of DeriveSK run by $\mathcal{O}_{\mathsf{sk}'}$ with '**return** $\mathsf{sk}' = g\hat{g}$' where $\hat{g}$ is obtained from the matching y' entry in List. This ensures that sk' output from the oracle still passes Check when called on a corresponding public key obtained from $\mathcal{O}_{\mathsf{pk}'}$. As both sk' and h are uniformly sampled from the same space, the two games are indistinguishable. Hence

$$\Pr[\mathcal{G}_1 = 1] = \Pr[\mathcal{G}_0 = 1]$$

We then construct an adversary $\mathcal{B}$ for the snPRF-OGADH game from an adversary that is able to win at $\mathcal{G}_1$. That is, break msKS security of ARKG.

The adversary $\mathcal{B}$ gets the challenge y from its snPRF-OGADH game. It sets up the game as described except it sets $\mathsf{sk} \leftarrow \perp$, $\mathsf{pk} \leftarrow y$, and the label of KDF_1 to be the challenge label x from its own game. It challenges $\mathcal{A}$ to create a forgery on y and is able to answer oracle queries honestly. Then, $\mathcal{A}$ outputs the tuple $(\mathsf{sk}^\star, \mathsf{pk}^\star, \mathsf{cred}^\star)$, from which $\mathcal{B}$ can create a successful answer to the snPRF-OGADH challenge $(x_0, y, \hat{y}, \tau^\star)$. It extracts y' from $\mathsf{cred}^\star$ and uses the single *OGADH* oracle query in the snPRF-OGADH game to get $\tau \leftarrow \mathsf{KDF}_1(g' \star y')$. $\mathcal{B}$ can then compute the secret key as $\mathbf{s} = \mathsf{sk}' - \mathbf{h}$. With knowledge of s it is trivial for $\mathcal{B}$ to compute $\tau^\star = \mathsf{KDF}_1(g \star y')$ and make the comparison $\tau \overset{?}{=} \tau^\star$. If they are equal, then $b = 0$, otherwise $b = 1$. The event that $\mathcal{A}$ queries $y' = \hat{y}$ (which would cause the experiment to abort) happens with probability q_o/p where p is the size of G and q_o is the number of oracle queries made by $\mathcal{A}$. This probability is negligible as p is large.

Thus, the advantage of $\mathcal{A}$ in $\mathsf{Exp}^{\mathsf{msKS}}_{\mathsf{ARKG},\mathcal{A}}(\lambda)$ is bound by an adversary $\mathcal{B}$ against the snPRF-OGADH assumption, giving

$$\mathsf{Adv}^{\mathsf{msKS}}_{\mathsf{ARKG},\mathcal{A}}(\lambda) \leqslant \frac{p - q_o}{p} \cdot \mathsf{Adv}^{\mathsf{snprf-ogadh}}_{G,\mathcal{B}}(\lambda)$$

If the snPRF-ODH assumption is hard in G then is msKS secure. □

Proof of Theorem 6.2. $\mathcal{G}_0$ is defined exactly by the experiment $\mathsf{Exp}^{\mathsf{hsKS}}_{\mathsf{ARKG},\mathcal{A}}(\lambda)$. Thus

$$\Pr[\mathcal{G}_0 = 1] = \Pr\Big[\mathsf{Exp}^{\mathsf{hsKS}}_{\mathsf{ARKG},\mathcal{A}}(\lambda) = 1\Big]$$

Define $\mathcal{G}_1$ as $\mathcal{G}_0$ with the exception that the computation of pk' is replaced with '$\mathbf{\hat{a}} \leftarrow\!\!\$\; C_{k_1} \times \ldots \times C_{k_n}, \hat{g} \leftarrow \prod_{i=1}^{n} g_i^{a_i}, \mathsf{pk}' \leftarrow h \star (\hat{g} \star x_0)$'. The adversary $\mathcal{B}$ keeps an internal list List that contains elements of the form $(y', g', \hat{g})$. If $\mathcal{A}$ queries $\mathcal{O}_{\mathsf{sk}'}$ with cred $\ni y$ such that $y \in$ List then it replaces sk' DeriveSK run by $\mathcal{O}_{\mathsf{sk}'}$ with '**return** $\mathsf{sk}' = g\hat{g}$' where $\hat{g}$ is obtained from the matching y' entry in List. This ensures that sk' output from the oracle still passes Check when called on a corresponding public key obtained from $\mathcal{O}_{\mathsf{pk}'}$. As both sk' and h are uniformly sampled from the same space, the two games are indistinguishable. Hence

$$\Pr[\mathcal{G}_1 = 1] = \Pr[\mathcal{G}_0 = 1]$$

We then construct an adversary $\mathscr{B}$ for the GAIP game from an adversary that is able to win at $\mathscr{G}_1$, that is, break hsKS security of ARKG. The adversary $\mathscr{B}$ sets up the game as described in $\mathscr{G}_1$, except that on line 2 it replaces '$(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KeyGen}$' with '$\mathsf{pk} \leftarrow y, \mathsf{sk} \leftarrow \bot$', where y is $\mathscr{B}$'s challenge from the GAIP game. The challenger $\mathscr{B}$ chooses one query to the oracle where it guesses $\mathscr{A}$ will use the derived pk' as part of its forgery. For this query, it can answer oracles calls to $\mathscr{O}_{\mathsf{pk}'}$ using the GAIP challenge y, and cannot answer $\mathscr{O}_{\mathsf{sk}'}$ queries. In the event this oracle is queried, the experiment aborts. For all other queries, $\mathscr{B}$ can answer oracle queries made by $\mathscr{A}$ as $\mathscr{B}$ generates the ephemeral key pair (g', y') and can extract the value r from List.

Then, $\mathscr{B}$ waits for $\mathscr{A}$ to output a successful forgery sk' with credential $\mathsf{cred}^\star$. Using y' from $\mathsf{cred}^\star$, $\mathscr{B}$ is able to locate y' in List and find the corresponding g'. This allows $\mathscr{B}$ to recompute ck from $\mathsf{KDF}_1(g' \star y)$ and compute an s such that $\prod_{i=1}^{n} g_i^{s_i} \star x_o = y$ as $\mathbf{s} = \mathsf{sk}' - \mathbf{h}$. $\mathscr{B}$ is guaranteed that $y' \in \mathsf{List}$ as a successful forgery in the hsKS game requires $(\mathsf{pk}^\star, \mathsf{cred}^\star) \in \mathsf{PKList}$ which can only happen if $\mathscr{B}$ generated y' during an oracle call from $\mathscr{A}$. However, the simulation fails if $\mathscr{A}$ queries $\mathscr{O}_{\mathsf{sk}'}$ on pk' that embeds the GAIP challenge, which happens with probability $1/q_o$. Thus, the advantage of $\mathscr{A}$ in $\mathsf{Exp}^{\mathsf{hsKS}}_{,\mathscr{A}}(\lambda)$ is bound by an adversary $\mathscr{B}$ against the (restricted) Group Action Inverse Problem assumption, giving

$$\mathsf{Adv}^{\mathsf{hwKS}}_{\mathsf{ARKG},\mathscr{A}}(\lambda) \leqslant \frac{1}{q_o} \mathsf{Adv}^{\mathsf{gaip}}_{G,\mathscr{B}}(\lambda)$$

By assumption, the GAIP problem is hard in (G, X) and thus is hsKS secure.

$\square$

Simulatable Key Derivation. We also achieve this simulable public key derivation property much like in our construction in Figure 17. The only different to note is that we are not given a explicit description of the group G rather than just a generating set. Since this generating set would naturally generate the group, then for any element $y \in X$ then there is a representation of a group element $g \in G$ as a sequence of operations $\prod g_i^{e_i}$ such that $y = g \star x$.

VII.5.4 Public Key Encryption from Group Actions

We adopt the public-key encryption schemes seen in their isogeny analogues and rephrase it in the context of cryptographic group actions. The algorithms for the scheme is given in Figure 19. In the description of these algorithms, the function $\mathscr{H} : X \to \{0,1\}^\lambda$ represents a hash function on the set X. By rephrasing the security proofs given by Stolbunov [Sto09] in the context of cryptographic group actions, then one can show that for either an effective or a restricted group action the described PKE is $\mathsf{IND\text{-}CPA}$-secure assuming the GA − DDH-assumption. In the algorithm specifications below we implicitly describe the setup algorithm that outputs a description of a set X, group G and hash function $\mathscr{H}$.

Depending on the type of cryptographic group action that is available, the ARKG schemes mentioned in §VII.5.2 and §VII.5.3 can be combined with the aforementioned PKE to obtain the key-private security property using the generic methods from Section VII.4. In particular, we obtain an $\mathsf{IND\text{-}IK\text{-}CPA}$-secure key-private public key encryption scheme.

```
KeyGen(pp)
1 : $g \leftarrow\!\!\$\; G$
2 : $y \leftarrow g \star x_0$
3 : $\mathsf{sk} := g$
4 : $\mathsf{pk} := y$
5 : return $(\mathsf{sk}, \mathsf{pk})$
```

```
Encrypt(pp, pk = $y$, m)
1 : $h \leftarrow\!\!\$\; G$
2 : $y' \leftarrow h \star x_0$
3 : $z \leftarrow h \star y$
4 : $c \leftarrow \mathsf{m} \oplus \mathscr{H}(z)$
5 : $\mathsf{c} := (y', c)$
6 : return c
```

```
Decrypt(pp, sk = $g$, c = $(y', c)$)
1 : $z \leftarrow g \star y'$
2 : $\mathsf{m} \leftarrow c \oplus \mathscr{H}(z)$
3 : return m
```

Figure 19: Algorithms for the public key encryption scheme from cryptographic group actions based on [Sto09].

Corollary VII.23. *Using the generic methods from Section VII.3, combining the restricted effective group action construction for PKE in Figure 19 with our ARKG construction in Figure 18 yields a UPKE that is* IND-CR-CPA *secure based on restricted effective group actions.*

Remark VII.24. *We note that our construction does not need any rejection sampling as seen in other CSIDH based constructions such as the signature scheme SeaSign [DFG19b]. This is because at no point are the secret vectors made publicly available which is the source of the leakage in the context of SeaSign. Additionally, the secret keys grows in size as they get updated. Hence, both encryption and decryption at each epoch becomes more expensive.*

VII.5.5 Restricting to the CSIDH group action

As mentioned in §VII.2.1 we have a group action from isogenies. As along as you have a complete description of the underlying class group then one can apply this to the ARKG construction of §VII.5.2 and obtain a UPKE. This construction is very similar to the one that was outlined in [EJKM22] since one needs to know the underlying class group structure in order to successfully execute the protocol. This class group computation requires an algorithm that has classical subexponential complexity. While we could treat this as precomputation as done for the CSIDH-512 parameter [BKV19], this would require a lot more computing resources that to do this for the higher CSIDH parameters. In addition, the recent quantum security analysis of these isogeny based group action protocols has come into question and it looks like the parameters would need to be scaled. In recent work [CSCDJRH22] parameters were suggested that better reflect this analysis. The size of these parameters make it classically infeasible to do these class group computations.

On the other hand as alluded earlier, the action used in the CSIDH key exchange protocol gives a restricted effective group action. Hence, one can apply the ARKG construction of §VII.5.3 to obtain a UPKE. This has a notable advantage of not requiring us to compute the whole class group and means that we can scale the parameters in line with the mentioned quantum analyses. Thus making it more practical than the other constructions. To achieve this, we rely on the heuristic that the ideal classes used in CSIDH actually cover the class group. No generic proof of this fact is known but the original proposal for CSIDH gave evidence to suggest it could be true [CLM$^+$18]. At least experimentally, this is known for the CSIDH-512 through the expensive class group computations [BKV19]. However, as mentioned in Remark VII.24, the keys grow in size at each iteration thus making encryption and decryption algorithms more expensive. This would be an issue in CSIDH since this algorithm are already not particularly efficient and applying these algorithms to a key that is quite large size would make it a lot slower. With realistic parameters in mind, only a few hundred key updates could be made before the encryption algorithms to become too slow for practical use.

The inclusion of the hash function in the generic construction of the KP-PKE is sufficient to show IND-CPA security for an arbitrary cryptographic group action but is not necessary in the context of the specific action used in CSIDH. We can replace the public-key encryption scheme described in §VII.5.4 with either public-key encryption scheme SiGamal [MOT20] or SimS [FP21b]. In these schemes, the use of a hash function is replaced with torsion point images. The inclusion of torsion point images in isogeny-based cryptosystems such as SIDH [DFJP14] have recently been detrimental to their security [CD23, MMP$^+$23, Rob23]. In particular, these attacks require the knowledge of torsion point images on a basis of the torsion subgroup as well as its degree. These two requirements seem to be necessary in order to run the attack and is exactly what we have in SIDH. However in SiGamal/SimS, we are only given one torsion point image rather than on a full basis. Also, the degree of the isogeny used in CSIDH/SiGamal/SimS is not publicly revealed unlike its SIDH counterpart. This makes running the attack rather difficult especially with the scaled parameters. As a light remark, the later of these points is similar to the countermeasure to the attacks on SIDH [FMP23] mentioned in Chapter IV.

In addition, the parameters proposed in SiGamal need to be scaled in line with the quantum security analyses mentioned previously. We leave this as a future research direction.

VII.6 Conclusion and Future Work

In this work we revisit updatable public key encryption from the perspective of cryptographic group actions. We identified in the literature that updatable public key encryption had only been explored from effective group actions through the isogeny specific construction by Eaton et al. [EJKM22]. We propose constructions from both effective and also restricted effective group actions. This significantly improves that of prior construction and means that it can be scaled to necessary parameters for security.

To achieve this, we first developed a generic technique for building UPKE from other cryptographic primitives. In particular, we show that if an ARKG scheme that is PK-unlinkable and KP-PKE is IND-IK-CPA secure, then we can construct a UPKE that is IND-CR-CPA secure. Additionally we provide a generic technique for building KP-PKE. In particular, we show that if an ARKG scheme that is PK-unlinkable and PKE is IND-CPA (IND-CCA) secure, then we can construct a KP-PKE that is IND-IK-CPA (IND-IK-CCA) secure. Thus achieving a UPKE from an ARKG and a PKE. Putting it into context of cryptographic group actions, we construct an ARKG scheme from both effective and restricted effective group actions. We prove its security using a new well motivated assumption called the PRF-OGADH assumption. This assumption is motivated from analogous definitions in the discrete logarithm setting and relate its hardness to other established assumptions within the realm of cryptographic group actions. We also use a generic PKE construction for cryptographic group actions which when combined with our ARKG scheme gives us a UPKE.

VII.6.1 Future Work

We identify a few research questions that we believe to be of interest.

- Our UPKE construction only achieves IND-CR-CPA security and did not explore the option of obtaining IND-CR-CCA security which would add an extra level of security to our constructions. We leave this as future work.
- We believe that the introduction of the assumption could have wider reaching applications outside the remit of our purposes. In addition, we only provide a small sample of security reductions of the various definitions of PRF-OGADH security. The details of the other reductions is left as future work
- In the construction of the KP-PKE from group actions mentioned in Section VII.5.4, the extra functionality that we impose on the PKE through an extra iteration of the ARKG scheme is sufficient to obtain the key-private security but we note that it might not be necessary. Classical discrete logarithm based PKEs such as ElGamal automatically have this key-private property [BBDP01b]. Given their similarity, we believe that the group action based PKE also has the key-private property without the need to use ARKG again. This would be improve the efficiency of the protocol. We leave the details of this as future work.
- Some of the proofs given here rely on the random oracle model. Hence if we wish to treat this as a post-quantum scheme then one would require proofs that utilise the quantum random oracle model. We leave this as a open problem to figure out these details.

Chapter VIII

Conclusion

In this final chapter, we recall the main contributions of this thesis as well as discuss some open problems and research that could prove useful in the context of isogeny-based cryptography.

Commitment scheme construction. The first contribution of the thesis came in Chapter V and consists of a provably secure commitment scheme based on supersingular isogeny graphs. This is the first such commitment scheme to be presented in the isogeny literature. The idea is inspired by the approach taken in the construction of the CGL hash function [CLG09]. In particular, to commit to a message string m one goes on a walk in some supersingular ℓ-isogeny graph. The paths direction within the graph is dictated by the message itself. After proceeding with this walk, one goes on a further walk whose path is dictated by a random string r. The culmination of this walk is a supersingular elliptic curve which serves as the commitment of the message m. When opening the commitment one simply the strings m, r and check that the committed curve is the same as that from what one would get by going on the walk with this m, r as described above.

At first glance, this idea looks very similar to a generic construction of a commitment scheme from a cryptographic hash function [Sma15, Chapter 20.2]. However, the generic construction requires the random oracle model in order to prove its security. In this construction we are able to prove its security in the standard model and show that our construction is information-theoretically hiding and computationally binding assuming the hardness of the one endomorphism problem. An essential component in proving the hiding security of our commitment scheme was the idea of the mixing constant. The definition of the mixing constant, in the context of the supersingular ℓ-isogeny graph, is the smallest integer m such that any two supersingular elliptic curves are connected in the supersingular ℓ-isogeny graph by a non-backtracking path of length m. There are numerous graph theoretic results of this type but one does not restrict themselves to solely non-backtracking paths so prior results could not be applied in this context. Through some experimental results, we conjecture a concrete upper bound for this mixing constant for any supersingular ℓ-isogeny graph (see Conjecture V.13 and V.17). We were not able to prove this conjecture within the time of completing this thesis, so one interesting open problem would be to try and see if this conjecture could be proven.

Another interesting open problem that is certainly worth exploring is the problem of constructing a homomorphic commitment scheme. The potential of such a homomorphic construction could be extremely interesting for more advanced applications that use commitment schemes as a building block.

Finding twin-smooth integers. The second contribution of the thesis came in Chapter VI in which the problem of finding twin-smooth integers is explored. This has recently been an interesting problem to explore within the of parameter setup of certain isogeny cryptosystems, such as B-SIDH and SQISign. In particular, we explore the CHM algorithm to find almost all B-smooth twins for some smoothness bound B. This constructive algorithm consists of starting with an initial set of B-smooth twins, namely $\{1, \cdots, B-1\}$, and continually include new B-smooth twins to this set until the algorithm terminates (which it must do since the number of B-smooth twins is finite). In particular, one computes the expression $(r(s+1))/((r+1)s)$ for r and s in the set and $r < s$ and if it can be reduced down to $t/(t+1)$ whereby t is not in the test, then we include t in this set. The original authors of the CHM algorithm [CHM13] ran this algorithm with a smoothness bound of $B = 200$. Their run of the algorithm took two weeks of computation before it terminated. Our contribution to this is introducing a bunch of optimisations in order to run this algorithm much faster. In particular, our run with $B = 200$ terminated in only 15 minutes. Subsequently, we ran the full algorithm for $B = 547$ which took between 1 and 2 days to run – the largest twin it found was 122 bits.

Despite these improvements, the CHM algorithm could not be utilised to find cryptographic sized twins suitable for applications to isogeny-based cryptography. The necessary smoothness bound B needed to run the algorithm and such twin large twins is not feasible to do using the current computing resources. Having said this, we utilised the smaller twins that were found with our runs and evaluated them on the polynomials $p_n(x)$ and $p_{i,j}(x)$ in order to find suitable parameters for SQISign. In particular, we present the first practical parameters that achieve more than 128-bits of classical security.

Additionally, we present a new probabilistic method for finding twin smooth integers. The idea utilises the extended Euclidean algorithm over polynomial rings in a similar way as done in the corresponding method that works over the integers. Moreover, this new method generalises the known methods that are based on polynomial evaluation, namely $r = x^n - 1$ and the PTE solutions. We give some evidence to suggest that there are some choice of polynomials that could be utilised to best smoother twins than that achieved by using the PTE solutions. We leave the problem of practically find cryptographic instances with this method as future work.

There are a few other open problems that are of interest to explore. The main one being to answer the question of why the CHM algorithm finds almost all twins for a given smoothness bound. Currently, there is no real justification for this other than experimental results of the algorithm and comparing it to a Pell equation computation when it can be done.

Updatable public-key encryption construction. The third and final contribution of this thesis came in Chapter VII whereby we construct a UPKE scheme using cryptographic group actions. The basis for the construction uses a generic a new cryptographic primitive called ARKG [FGK+20]. The idea behind ARKG is to be able to derive a new public-key pair from an existing public-key pair is a way so that the two key pairs are unlinkable and the derivation of the public-key and the private-key can be done asynchronously. While this was initially motivated as a protocol for use in the WebAuthn account recovery process, as a cryptographic primitive it has wider applications. In particular, we present a generic construction of an UPKE that uses an ARKG scheme in combination with a key-private public-key encryption scheme. Furthermore, we instantiate our scheme with cryptographic group actions by providing a construction of an ARKG scheme from these group actions. The framework for this construction is inspired with discrete logarithm construction given in the seminal work [FGK$^+$20].

While there are other UPKE schemes based on cryptographic group action in the literature [EJKM22], we are the first one that are able to construct such UPKE's from restricted effective group actions. In particular, our scheme can be instantiated using the setup given in CSIDH whereby an expensive precomputation of the class group is not needed. This means that our scheme scales much better than the previous approach as we increase the security parameter. Having said this, with each epoch in our UPKE, we increase the size of the private key. Hence this makes encryption and decryption more expensive as the epoch increases. As a result, it is expected that it would only be able to handle around 100 iterations before grinding to a halt. This means that it will most likely not be used in a practical setting and just be there for theoretical interest.

There are a number of open problems that arise from this work as mentioned in Section VII.6.1. An additional open problem is whether ARKG can be used as a tool to construct other cryptographic primitives other than UPKE.

Future of isogeny-based cryptography. Despite the setback that was caused by the recent attacks on SIDH, isogeny-based cryptography has a bright future with CSIDH and SQISign making the main headlines for the future years. The fundamental problem of computing an isogeny between supersingular elliptic curves is believed to be hard. In comparison to other post-quantum approaches, isogeny-based cryptosystems offer small keys and short signatures at a cost of a slower performance. For certain applications this is something that could be beneficial. One application that would be useful to explore is in the setting of Trusted Platform Modules (TPM) where having small key sizes appears to be more beneficial. To truly realise this, work on group signatures would need to be further developed to yield a particular strong application but SQISign as a signature scheme on its own would be quite interesting when incoporated into a TPM. Many other applications can be thought about and identifying these will be an important future problem for the development of isogeny-based cryptography. Other open problems include: improving the efficiency of existing cryptosystems, designing more protocols to serve isogeny friendly applications and also conduct security analysis and cryptanalysis of existing cryptosystems to assess their confidence.

Bibliography

[ABB$^+$17] Nicolas Aragon, Paulo Barreto, Slim Bettaieb, Loïc Bidoux, Olivier Blazy, Jean-Christophe Deneuville, Philippe Gaborit, Shay Gueron, Tim Guneysu, Carlos Aguilar Melchor, et al. BIKE: bit flipping key encapsulation. 2017.

[ACDT20] Joël Alwen, Sandro Coretti, Yevgeniy Dodis, and Yiannis Tselekounis. Security analysis and improvements for the ietf mls standard for group messaging. In *CRYPTO*, 2020.

[ACNL$^+$19] Sarah Arpin, Catalina Camacho-Navarro, Kristin Lauter, Joelle Lim, Kristina Nelson, Travis Scholl, and Jana Sotáková. Adventures in supersingularland. *arXiv preprint arXiv:1909.07779*, 2019.

[ACVCD$^+$19] Gora Adj, Daniel Cervantes-Vázquez, Jesús-Javier Chi-Domínguez, Alfred Menezes, and Francisco Rodríguez-Henríquez. On the cost of computing isogenies between supersingular elliptic curves. In *Selected Areas in Cryptography–SAC 2018: 25th International Conference, Calgary, AB, Canada, August 15–17, 2018, Revised Selected Papers*, pages 322–343. Springer, 2019.

[ADFMP20] Navid Alamati, Luca De Feo, Hart Montgomery, and Sikhar Patranabis. Cryptographic group actions and applications. In Shiho Moriai and Huaxiong Wang, editors, *Advances in Cryptology – ASIACRYPT 2020*, pages 411–439, Cham, 2020. Springer International Publishing.

[AEK$^+$22] Michel Abdalla, Thorsten Eisenhofer, Eike Kiltz, Sabrina Kunzweiler, and Doreen Riepel. Password-authenticated key exchange from group actions. In *Advances in Cryptology–CRYPTO 2022: 42nd Annual International Cryptology Conference, CRYPTO 2022, Santa Barbara, CA, USA, August 15–18, 2022, Proceedings, Part II*, pages 699–728. Springer, 2022.

[AJJS19] Reza Azarderakhsh, Amir Jalali, David Jao, and Vladimir Soukharev. Practical supersingular isogeny group key agreement. *Cryptology ePrint Archive*, 2019.

[AJL18] Reza Azarderakhsh, David Jao, and Christopher Leonardi. Post-quantum static-static key agreement using multiple protocol instances. In *Selected Areas in Cryptography–SAC 2017: 24th International Conference, Ottawa, ON, Canada, August 16-18, 2017, Revised Selected Papers 24*, pages 45–63. Springer, 2018.

[Alo86] Noga Alon. Eigenvalues and expanders. *Combinatorica*, 6(2):83–96, 1986.

[Bas23] Andrea Basso. A post-quantum round-optimal oblivious prf from isogenies. Cryptology ePrint Archive, Paper 2023/225, 2023. `https://eprint.iacr.org/2023/225`.

[BBD$^+$22] Jeremy Booher, Ross Bowden, Javad Doliskani, Tako Boris Fouotsa, Steven D Galbraith, Sabrina Kunzweiler, Simon-Philipp Merz, Christophe Petit, Benjamin Smith, Katherine E Stange, et al. Failing to hash into supersingular isogeny graphs. *arXiv preprint arXiv:2205.00135*, 2022.

[BBDP01a] Mihir Bellare, Alexandra Boldyreva, Anand Desai, and David Pointcheval. Key-privacy in public-key encryption. In Colin Boyd, editor, *ASIACRYPT*, 2001.

[BBDP01b] Mihir Bellare, Alexandra Boldyreva, Anand Desai, and David Pointcheval. Key-privacy in public-key encryption. In *Advances in Cryptology—ASIACRYPT 2001: 7th International Conference on the Theory and Application of Cryptology and Information Security Gold Coast, Australia, December 9–13, 2001 Proceedings 7*, pages 566–582. Springer, 2001.

[BBJ$^+$08] Daniel J Bernstein, Peter Birkner, Marc Joye, Tanja Lange, and Christiane Peters. Twisted edwards curves. In *International Conference on Cryptology in Africa*, pages 389–405. Springer, 2008.

[BBR18] Karthikeyan Bhargavan, Richard Barnes, and Eric Rescorla. TreeKEM: Asynchronous Decentralized Key Management for Large Dynamic Groups A protocol proposal for Messaging Layer Security (MLS). Research report, Inria Paris, May 2018.

[BCC$^+$23] Andrea Basso, Giulio Codogni, Deirdre Connolly, Luca De Feo, Tako Boris Fouotsa, Guido Maria Lido, Travis Morrison, Lorenz Panny, Sikhar Patranabis, and Benjamin Wesolowski. Supersingular curves you can trust, 2023.

[BCK96] Mihir Bellare, Ran Canetti, and Hugo Krawczyk. Keying hash functions for message authentication. In *Advances in Cryptology – CRYPTO '96*, pages 1–15. Springer Berlin Heidelberg, 1996.

[BDF21] Jeffrey Burdges and Luca De Feo. Delay encryption. In *Advances in Cryptology–EUROCRYPT 2021: 40th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, October 17–21, 2021, Proceedings, Part I*, pages 302–326. Springer, 2021.

[BDGJ20] Colin Boyd, Gareth T. Davies, Kristian Gjøsteen, and Yao Jiang. Fast and secure updatable encryption. In *CRYPTO*, 2020.

[BDK$^+$18] Joppe Bos, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, John M Schanck, Peter Schwabe, Gregor Seiler, and Damien Stehlé. Crystals-kyber: a cca-secure module-lattice-based kem. In *2018 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 353–367. IEEE, 2018.

[BDL$^+$18] Carsten Baum, Ivan Damgård, Vadim Lyubashevsky, Sabine Oechsner, and Chris Peikert. More efficient commitments from structured lattice assumptions. In Dario Catalano and Roberto De Prisco, editors, *Security and Cryptography for Networks*, pages 368–385, Cham, 2018. Springer International Publishing.

[Ber04] Daniel J Bernstein. How to find smooth parts of integers. *URL: http://cr. yp. to/papers. html# smoothparts. ID 201a045d5bb24f43f0bd0d97fcf5355a. Citations in this document*, 20, 2004.

[BFGJ17] Jacqueline Brendel, Marc Fischlin, Felix Günther, and Christian Janson. Prf-odh: Relations, instantiations, and impossibility results. In *CRYPTO*, 2017.

[BFLS20] Daniel J. Bernstein, Luca De Feo, Antonin Leroux, and Benjamin Smith. Faster computation of isogenies of large prime degree. Cryptology ePrint Archive, Report 2020/341, 2020. `https://eprint.iacr.org/2020/341`.

[BGB04] Nikita Borisov, Ian Goldberg, and Eric Brewer. Off-the-record communication, or, why not to use pgp. In *Workshop on Privacy in the Electronic Society*. ACM, 2004.

[BHH$^+$15] Daniel J Bernstein, Daira Hopwood, Andreas Hülsing, Tanja Lange, Ruben Niederhagen, Louiza Papachristodoulou, Michael Schneider, Peter Schwabe, and Zooko Wilcox-O'Hearn. SPHINCS: practical stateless hash-based signatures. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 368–397. Springer, 2015.

[BHKL13] Daniel J Bernstein, Mike Hamburg, Anna Krasnova, and Tanja Lange. Elligator: Elliptic-curve points indistinguishable from uniform random strings. In *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*, pages 967–980, 2013.

[BHL$^+$22] Jan Buzek, Junaid Hasan, Jason Liu, Michael Naehrig, and Anthony Vigil. Finding twin smooth integers by solving pell equations. *arXiv preprint arXiv:2211.04315*, 2022.

[Bin18] Timo Bingmann. TLX: Collection of sophisticated C++ data structures, algorithms, and miscellaneous helpers, 2018. `https://panthema.net/tlx`, retrieved Oct. 7, 2020.

[BJS14] Jean-François Biasse, David Jao, and Anirudh Sankar. A quantum algorithm for computing isogenies between supersingular elliptic curves. In *Progress in Cryptology–INDOCRYPT 2014: 15th International Conference on Cryptology in India, New Delhi, India, December 14-17, 2014, Proceedings*, pages 428–442. Springer, 2014.

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

[BKV19] Ward Beullens, Thorsten Kleinjung, and Frederik Vercauteren. Csi-fish: Efficient isogeny based signatures through class group computations. In Steven D. Galbraith and Shiho Moriai, editors, *Advances in Cryptology – ASIACRYPT 2019*, pages 227–247, Cham, 2019. Springer International Publishing.

[BKW20] Dan Boneh, Dmitry Kogan, and Katharine Woo. Oblivious pseudorandom functions from isogenies. In *Advances in Cryptology–ASIACRYPT 2020: 26th International Conference on the Theory and Application of Cryptology and Information Security, Daejeon, South Korea, December 7–11, 2020, Proceedings, Part II 26*, pages 520–550. Springer, 2020.

[BLMR13] Dan Boneh, Kevin Lewi, Hart William Montgomery, and Ananth Raghunathan. Key homomorphic prfs and their applications. In *CRYPTO*, 2013.

[BLP03] Peter Borwein, Petr Lisoněk, and Colin Percival. Computational investigations of the prouhet-tarry-escott problem. *Mathematics of computation*, 72(244):2063–2070, 2003.

[BLSV18] Zvika Brakerski, Alex Lombardi, Gil Segev, and Vinod Vaikuntanathan. *Anonymous IBE, Leakage Resilience and Circular Security from New Assumptions*. 2018.

[Blu83] Manuel Blum. Coin flipping by telephone a protocol for solving impossible problems. *ACM SIGACT News*, 15(1):23–27, 1983.

[Brö09] Reinier Bröker. Constructing supersingular elliptic curves. *J. Comb. Number Theory*, 1(3):269–273, 2009.

[BRO$^+$22] B. Beurdouche, E. Rescorla, E. Omara, S. Inguva, A. Kwon, and A. Duric. draft-ietf-mls-architecture-latest. `https://messaginglayersecurity.rocks/mls-architecture/draft-ietf-mls-architecture.html`, Accessed 2022.

[BS06] William D Banks and Igor E Shparlinski. Integers with a large smooth divisor. *arXiv preprint math/0601460*, 2006.

[BS11] Gaetan Bisson and Andrew V Sutherland. Computing the endomorphism ring of an ordinary elliptic curve over a finite field. *Journal of Number Theory*, 131(5):815–831, 2011.

[BS20] Xavier Bonnetain and André Schrottenloher. Quantum security analysis of CSIDH. In *Advances in Cryptology–EUROCRYPT 2020: 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, May 10–14, 2020, Proceedings, Part II 30*, pages 493–522. Springer, 2020.

[BSC$^+$22] Giacomo Bruno, Maria Corte-Real Santos, Craig Costello, Jonathan Komada Eriksen, Michael Naehrig, Michael Meyer, and Bruno Sterner. Cryptographic smooth neighbors. *Cryptology ePrint Archive*, 2022.

[Buc37] AA Buchstab. Asymptotic estimates of a general number-theoretic function. *Mat. Sb*, 44(2):1239–1246, 1937.

[Cal13] Timothy Caley. *The Prouhet-Tarry-Escott problem*. PhD thesis, University of Waterloo, 2013.

[CD20] Wouter Castryck and Thomas Decru. Csidh on the surface. In *Post-Quantum Cryptography: 11th International Conference, PQCrypto 2020, Paris, France, April 15–17, 2020, Proceedings*, pages 111–129. Springer, 2020.

[CD23] Wouter Castryck and Thomas Decru. An efficient key recovery attack on sidh. In *Advances in Cryptology – EUROCRYPT 2023*, pages 423–447, Cham, 2023. Springer Nature Switzerland.

[CDG$^+$17] Melissa Chase, David Derler, Steven Goldfeder, Claudio Orlandi, Sebastian Ramacher, Christian Rechberger, Daniel Slamanig, and Greg Zaverucha. Post-quantum zero-knowledge and signatures from symmetric-key primitives. In *Proceedings of the 2017 acm sigsac conference on computer and communications security*, pages 1825–1842, 2017.

[CFSY96] Ronald Cramer, Matthew Franklin, Berry Schoenmakers, and Moti Yung. Multi-authority secret-ballot elections with linear work. In Ueli Maurer, editor, *Advances in Cryptology — EUROCRYPT '96*, pages 72–83, Berlin, Heidelberg, 1996. Springer Berlin Heidelberg.

[CH17] Craig Costello and Huseyin Hisil. A simple and compact algorithm for sidh with arbitrary degree isogenies. In *Advances in Cryptology–ASIACRYPT 2017: 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part II 23*, pages 303–329. Springer, 2017.

[CH21] JB Conrey and MA Holmstrom. Smooth values of quadratic polynomials. *Experimental Mathematics*, 30(4):447–452, 2021.

[CHK04] Ran Canetti, Shai Halevi, and Jonathon Katz. Chosen-ciphertext security from identity-based encryption. In *EUROCRYPT 2004*, 2004.

[CHKP10] David Cash, Dennis Hofheinz, Eike Kiltz, and Chris Peikert. Bonsai trees, or how to delegate a lattice basis. In *EUROCRYPT 2010*, 2010.

[CHM13] J. B. Conrey, M. A. Holmstrom, and T. L. McLaughlin. Smooth neighbors. *Experimental Mathematics*, 22(2):195–202, 2013.

[CJL+17] Craig Costello, David Jao, Patrick Longa, Michael Naehrig, Joost Renes, and David Urbanik. Efficient compression of SIDH public keys. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 679–706. Springer, 2017.

[CJS14] Andrew Childs, David Jao, and Vladimir Soukharev. Constructing elliptic curve isogenies in quantum subexponential time. *Journal of Mathematical Cryptology*, 8(1):1–29, 2014.

[CK01] Ran Canetti and Hugo Krawczyk. Analysis of key-exchange protocols and their use for building secure channels. In Birgit Pfitzmann, editor, *Advances in Cryptology — EUROCRYPT 2001*, pages 453–474, Berlin, Heidelberg, 2001. Springer Berlin Heidelberg.

[CLG09] Denis X. Charles, Kristin E. Lauter, and Eyal Z. Goren. Cryptographic hash functions from expander graphs. *Journal of Cryptology*, 22(1):93–113, Jan 2009.

[CLM+18] Wouter Castryck, Tanja Lange, Chloe Martindale, Lorenz Panny, and Joost Renes. CSIDH: An efficient post-quantum commutative group action. In Thomas Peyrin and Steven Galbraith, editors, *Advances in Cryptology – ASIACRYPT 2018*, pages 395–427, Cham, 2018. Springer International Publishing.

[CMN21] Craig Costello, Michael Meyer, and Michael Naehrig. Sieving for twin smooth integers with solutions to the Prouhet-Tarry-Escott problem. In *Advances in Cryptology–EUROCRYPT 2021: 40th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, October 17–21, 2021, Proceedings, Part I*, pages 272–301. Springer, 2021.

[Cos20] Craig Costello. B-SIDH: supersingular isogeny Diffie-Hellman using twisted torsion. In *Advances in Cryptology–ASIACRYPT 2020: 26th International Conference on the Theory and Application of Cryptology and Information Security, Daejeon, South Korea, December 7–11, 2020, Proceedings, Part II 26*, pages 440–463. Springer, 2020.

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

[Cox89] David A Cox. Primes of the form $x^2 + ny^2$. 1989.

[CP01] Richard Crandall and Carl Pomerance. *Prime numbers*. Springer, 2001.

[CRSCS22] Maria Corte-Real Santos, Craig Costello, and Jia Shi. Accelerating the delfs–galbraith algorithm with fast subfield root detection. In Yevgeniy Dodis and Thomas Shrimpton, editors, *Advances in Cryptology – CRYPTO 2022*, pages 285–314, Cham, 2022. Springer Nature Switzerland.

[CS03] Ronald Cramer and Victor Shoup. Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. *SIAM Journal on Computing*, 33(1):167–226, 2003.

[CSCDJRH22] Jorge Chávez-Saab, Jesús-Javier Chi-Domínguez, Samuel Jaques, and Francisco Rodríguez-Henríquez. The SQALE of CSIDH: sublinear vélu quantum-resistant isogeny action with low exponents. *Journal of Cryptographic Engineering*, 12(3):349–368, 2022.

[Dam98] Ivan Damgård. Commitment schemes and zero-knowledge protocols. In *School organized by the European Educational Forum*, pages 63–86. Springer, 1998.

[DB66] NG De Bruijn. On the number of positive integers $\leq$ x and free of prime factors $> y$, ii. *Indag. Math*, 38:239–247, 1966.

[DEG17] Ashraf Darwish and Maged M El-Gendy. A new cryptographic voting verifiable scheme for e-voting system based on bit commitment and blind signature. *Int J Swarm Intel Evol Comput*, 6(158):2, 2017.

[Deu41] Max Deuring. Die typen der multiplikatorenringe elliptischer funktionenkörper. In *Abhandlungen aus dem mathematischen Seminar der Universität Hamburg*, volume 14, pages 197–272. Springer, 1941.

[DFDdSGF$^+$21] Luca De Feo, Cyprien Delpech de Saint Guilhem, Tako Boris Fouotsa, Péter Kutas, Antonin Leroux, Christophe Petit, Javier Silva, and Benjamin Wesolowski. Séta: Supersingular encryption from torsion attacks. In *Advances in Cryptology–ASIACRYPT 2021: 27th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 6–10, 2021, Proceedings, Part IV 27*, pages 249–278. Springer, 2021.

[DFDGZ23] Luca De Feo, Samuel Dobson, Steven D Galbraith, and Lukas Zobernig. Sidh proof of knowledge. In *Advances in Cryptology–ASIACRYPT 2022: 28th International Conference on the Theory and Application of Cryptology and Information Security, Taipei, Taiwan, December 5–9, 2022, Proceedings, Part II*, pages 310–339. Springer, 2023.

[DFG19a] Luca De Feo and Steven D Galbraith. Seasign: compact isogeny signatures from class group actions. In *Advances in Cryptology–EUROCRYPT 2019: 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19–23, 2019, Proceedings, Part III 38*, pages 759–789. Springer, 2019.

[DFG19b] Luca De Feo and Steven D. Galbraith. Seasign: Compact isogeny signatures from class group actions. In Yuval Ishai and Vincent Rijmen, editors, *Advances in Cryptology – EUROCRYPT 2019*, pages 759–789, Cham, 2019. Springer International Publishing.

[DFJP14] Luca De Feo, David Jao, and Jérôme Plût. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. *Journal of Mathematical Cryptology*, 8(3):209–247, 2014.

[DFKL$^+$20] Luca De Feo, David Kohel, Antonin Leroux, Christophe Petit, and Benjamin Wesolowski. Sqisign: compact post-quantum signatures from quaternions and isogenies. In *Advances in Cryptology–ASIACRYPT 2020: 26th International Conference on the Theory and Application of Cryptology and Information Security, Daejeon, South Korea, December 7–11, 2020, Proceedings, Part I 26*, pages 64–93. Springer, 2020.

[DFKS18] Luca De Feo, Jean Kieffer, and Benjamin Smith. Towards practical key exchange from ordinary isogeny graphs. In *Advances in Cryptology–ASIACRYPT 2018: 24th International Conference on the Theory and Application of Cryptology and Information Security, Brisbane, QLD, Australia, December 2–6, 2018, Proceedings, Part III 24*, pages 365–394. Springer, 2018.

[DFLLW23] Luca De Feo, Antonin Leroux, Patrick Longa, and Benjamin Wesolowski. New algorithms for the deuring correspondence. In Carmit Hazay and Martijn Stam, editors, *Advances in Cryptology – EUROCRYPT 2023*, pages 659–690, Cham, 2023. Springer Nature Switzerland.

[DFMPS19] Luca De Feo, Simon Masson, Christophe Petit, and Antonio Sanso. Verifiable delay functions from supersingular isogenies and pairings. In Steven D. Galbraith and Shiho Moriai, editors, *Advances in Cryptology – ASIACRYPT 2019*, pages 248–277, Cham, 2019. Springer International Publishing.

[DG16] Christina Delfs and Steven D Galbraith. Computing isogenies between supersingular elliptic curves over $\mathbb{F}_p$. *Designs, Codes and Cryptography*, 78(2):425–440, 2016.

[DG17a] Nico Döttling and Sanjam Garg. From selective ibe to full ibe and selective hibe. In Yael Kalai and Leonid Reyzin, editors, *Theory of Cryptography*, pages 372–408, Cham, 2017. Springer International Publishing.

[DG17b] Nico Döttling and Sanjam Garg. Identity-based encryption from the diffie-hellman assumption. In *CRYPTO*, 2017.

[DGL$^+$20] Samuel Dobson, Steven D Galbraith, Jason LeGrow, Yan Bo Ti, and Lukas Zobernig. An adaptive attack on 2-sidh. *International Journal of Computer Mathematics: Computer Systems Theory*, 5(4):282–299, 2020.

[DH76] Whitfield Diffie and Martin Hellman. New directions in cryptography. *IEEE transactions on Information Theory*, 22(6):644–654, 1976.

[DHK$^+$23] Julien Duman, Dominik Hartmann, Eike Kiltz, Sabrina Kunzweiler, Jonas Lehmann, and Doreen Riepel. Group action key encapsulation and non-interactive key exchange in the qrom. In *Advances in Cryptology–ASIACRYPT*

2022: 28th International Conference on the Theory and Application of Cryptology and Information Security, Taipei, Taiwan, December 5–9, 2022, Proceedings, Part II, pages 36–66. Springer, 2023.

[Dic30] Karl Dickman. On the frequency of numbers containing prime factors of a certain relative magnitude. *Arkiv for matematik, astronomi och fysik*, 22(10):A–10, 1930.

[DKL$^+$18] Léo Ducas, Eike Kiltz, Tancrede Lepoint, Vadim Lyubashevsky, Peter Schwabe, Gregor Seiler, and Damien Stehlé. Crystals-dilithium: A lattice-based digital signature scheme. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pages 238–268, 2018.

[DKW21] Yevgeniy Dodis, Harish Karthikeyan, and Daniel Wichs. Updatable public key encryption in the standard model. In *TCC*, 2021.

[DLRW23] Pierrick Dartois, Antonin Leroux, Damien Robert, and Benjamin Wesolowski. Sqisignhd: New dimensions in cryptography. Cryptology ePrint Archive, Paper 2023/436, 2023. `https://eprint.iacr.org/2023/436`.

[DPB17] Javad Doliskani, Geovandro C. C. F. Pereira, and Paulo S. L. M. Barreto. Faster cryptographic hash function from supersingular isogeny graphs. Cryptology ePrint Archive, Report 2017/1202, 2017. `https://eprint.iacr.org/2017/1202`.

[DPV19] Thomas Decru, Lorenz Panny, and Frederik Vercauteren. Faster seasign signatures through improved rejection sampling. In *Post-Quantum Cryptography: 10th International Conference, PQCrypto 2019, Chongqing, China, May 8–10, 2019 Revised Selected Papers 10*, pages 271–285. Springer, 2019.

[dQKL$^+$21] Victoria de Quehen, Péter Kutas, Chris Leonardi, Chloe Martindale, Lorenz Panny, Christophe Petit, and Katherine E Stange. Improved torsion-point attacks on sidh variants. In *Advances in Cryptology–CRYPTO 2021: 41st Annual International Cryptology Conference, CRYPTO 2021, Virtual Event, August 16–20, 2021, Proceedings, Part III 41*, pages 432–470. Springer, 2021.

[DR01] Joan Daemen and Vincent Rijmen. Reijndael: The advanced encryption standard. *Dr. Dobb's Journal: Software Tools for the Professional Programmer*, 26(3):137–139, 2001.

[DS05] Jintai Ding and Dieter Schmidt. Rainbow, a new multivariable polynomial signature scheme. In *International Conference on Applied Cryptography and Network Security*, pages 164–175. Springer, 2005.

[E$^+$98] Noam D Elkies et al. Elliptic and modular curves over finite fields and related computational issues. *AMS IP STUDIES IN ADVANCED MATHEMATICS*, 7:21–76, 1998.

[Edw07] Harold Edwards. A normal form for elliptic curves. *Bulletin of the American mathematical society*, 44(3):393–422, 2007.

[EHL$^+$18] Kirsten Eisenträger, Sean Hallgren, Kristin Lauter, Travis Morrison, and Christophe Petit. Supersingular isogeny graphs and endomorphism rings: reductions and solutions. In *Advances in Cryptology–EUROCRYPT 2018: 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Israel, April 29-May 3, 2018 Proceedings, Part III 37*, pages 329–368. Springer, 2018.

[EHL$^+$20] Kirsten Eisenträger, Sean Hallgren, Chris Leonardi, Travis Morrison, and Jennifer Park. Computing endomorphism rings of supersingular elliptic curves and connections to path-finding in isogeny graphs. *Open Book Series*, 4(1):215–232, 2020.

[EHM17] Kirsten Eisentraeger, Sean Hallgren, and Travis Morrison. On the hardness of computing endomorphism rings of supersingular elliptic curves. Cryptology ePrint Archive, Paper 2017/986, 2017. `https://eprint.iacr.org/2017/986`.

[EJKM22] Edward Eaton, David Jao, Chelsea Komlo, and Youcef Mokrani. Towards post-quantum key-updatable public-key encryption via supersingular isogenies. In *Selected Areas in Cryptography: 28th International Conference, Virtual Event, September 29–October 1, 2021, Revised Selected Papers*, pages 461–482. Springer, 2022.

[ElG85] Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE transactions on information theory*, 31(4):469–472, 1985.

[EPSV23] Jonathan Komada Eriksen, Lorenz Panny, Jana Sotáková, and Mattia Veroni. Deuring for the people: Supersingular elliptic curves with prescribed endomorphism ring in general characteristic. *Cryptology ePrint Archive*, 2023. `https://eprint.iacr.org/2023/106`.

[ETS] ISGQSC ETSI. Quantum-safe cryptography (qsc); quantum-safe algorithmic framework.

[FFK00] Michael Filaseta, Kevin Ford, and Sergei Konyagin. On an irreducibility theorem of a. schinzel associated with coverings of the integers. *Illinois Journal of Mathematics*, 44(3):633–643, 2000.

[FFK$^+$23] Luca De Feo, Tako Boris Fouotsa, Péter Kutas, Antonin Leroux, Simon-Philipp Merz, Lorenz Panny, and Benjamin Wesolowski. SCALLOP: scaling the CSI-FiSh. Cryptology ePrint Archive, Paper 2023/058, 2023. `https://eprint.iacr.org/2023/058`.

[FGK$^+$20] Nick Frymann, Daniel Gardham, Franziskus Kiefer, E. Lundberg, M. Manulis, and Dain Nilsson. Asynchronous remote key generation: An analysis of yubico's proposal for w3c webauthn. *ACM CCS*, 2020.

[FHK$^+$18] Pierre-Alain Fouque, Jeffrey Hoffstein, Paul Kirchner, Vadim Lyubashevsky, Thomas Pornin, Thomas Prest, Thomas Ricosset, Gregor Seiler, William Whyte, Zhenfei Zhang, et al. Falcon: Fast-fourier lattice-based compact signatures over ntru. *Submission to the NIST's post-quantum cryptography standardization process*, 36(5), 2018.

[FHKP13] Eduarda SV Freire, Dennis Hofheinz, Eike Kiltz, and Kenneth G Paterson. Non-interactive key exchange. In *Public-Key Cryptography–PKC 2013: 16th International Conference on Practice and Theory in Public-Key Cryptography, Nara, Japan, February 26–March 1, 2013. Proceedings 16*, pages 254–271. Springer, 2013.

[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, *Advances in Cryptology – EUROCRYPT 2023*, pages 282–309, Cham, 2023. Springer Nature Switzerland.

[FO99] Eiichiro Fujisaki and Tatsuaki Okamoto. Secure integration of asymmetric and symmetric encryption schemes. In *Annual International Cryptology Conference*, pages 537–554. Springer, 1999.

[FP21a] Tako Boris Fouotsa and Christophe Petit. Sheals and heals: isogeny-based pkes from a key validation method for sidh. In *Advances in Cryptology–ASIACRYPT 2021: 27th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 6–10, 2021, Proceedings, Part IV 27*, pages 279–307. Springer, 2021.

[FP21b] Tako Boris Fouotsa and Christophe Petit. SimS: a simplification of SiGamal. In *Post-Quantum Cryptography: 12th International Workshop, PQCrypto 2021, Daejeon, South Korea, July 20–22, 2021, Proceedings 12*, pages 277–295. Springer, 2021.

[FS87] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In Andrew M. Odlyzko, editor, *Advances in Cryptology — CRYPTO' 86*, pages 186–194, Berlin, Heidelberg, 1987. Springer Berlin Heidelberg.

[Gal99] Steven D Galbraith. Constructing isogenies between elliptic curves over finite fields. *LMS Journal of Computation and Mathematics*, 2:118–138, 1999.

[Gal12] Steven D. Galbraith. *Mathematics of Public Key Cryptography*. Cambridge University Press, USA, 1st edition, 2012.

[Gal20] Steven D. Galbraith. Talk given at the indian institute of technology kharagpur: “similarities and differences between diffie-hellman and isogeny crypto", August 2020.

[GGM86] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *Journal of the ACM*, 33(4):792–807, aug 1986.

[GL22] Steven D Galbraith and Yi-Fu Lai. Attack on sheals and heals: The second wave of gpst. In *Post-Quantum Cryptography: 13th International Workshop, PQCrypto 2022, Virtual Event, September 28–30, 2022, Proceedings*, pages 399–421. Springer, 2022.

[GPS17] Steven D. Galbraith, Christophe Petit, and Javier Silva. Identification protocols and signature schemes based on supersingular isogeny problems. In Tsuyoshi Takagi and Thomas Peyrin, editors, *Advances in Cryptology – ASIACRYPT 2017*, pages 3–33, Cham, 2017. Springer International Publishing.

[GPST16] Steven D. Galbraith, Christophe Petit, Barak Shani, and Yan Bo Ti. On the security of supersingular isogeny cryptosystems. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *Advances in Cryptology – ASIACRYPT 2016*, pages 63–91, Berlin, Heidelberg, 2016. Springer Berlin Heidelberg.

[GPV08] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *STOC*, page 197–206, 2008.

[Gro96] Lov K Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 212–219, 1996.

[GS23] Daniel Gardham and Bruno Sterner. Updatable public-key encryption: Generic constructions and instantiations from group actions. *Manuscript*, 2023.

[Hil86] Adolf Hildebrand. On the number of positive integers $\leq$ x and free of prime factors> y. *Journal of Number Theory*, 22(3):289–307, 1986.

[HL02] J. A. K. Horwitz and Ben Lynn. Toward hierarchical identity-based encryption. In *EUROCRYPT*, 2002.

[JAC$^+$17] David Jao, Reza Azarderakhsh, Matt Campagna, Craig Costello, Luca De Feo, Basil Hess, Amir Jalili, Brian Koziel, Brian Lamacchia, Patrick Longa, et al. SIKE: supersingular isogeny key encapsulation. 2017.

[JDF11] David Jao and Luca De Feo. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. In Bo-Yin Yang, editor, *Post-Quantum Cryptography*, pages 19–34, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[JMM19] Daniel Jost, Ueli Maurer, and Marta Mularczyk. Efficient ratcheting: Almost-optimal guarantees for secure messaging. In *Advances in Cryptology - EUROCRYPT*, 2019.

[JMV09] David Jao, Stephen D Miller, and Ramarathnam Venkatesan. Expander graphs based on GRH with an application to elliptic curve cryptography. *Journal of Number Theory*, 129(6):1491–1504, 2009.

[JS18] Joseph Jaeger and Igors Stepanovs. *Optimal Channel Security Against Fine-Grained State Compromise: The Safety of Messaging*. 2018.

[Kan97] Ernst Kani. The number of curves of genus two with elliptic differentials. 1997.

[KL20] Jonathan Katz and Yehuda Lindell. *Introduction to modern cryptography*. CRC press, 2020.

[KLM$^+$15] Daniel Kirkwood, Bradley C Lackey, John McVey, Mark Motley, Jerome A Solinas, and David Tuller. Failure is not an option: Standardization issues for post-quantum key agreement. In *Workshop on Cybersecurity in a Post-Quantum World*, page 21, 2015.

[KLPT14] David Kohel, Kristin Lauter, Christophe Petit, and Jean-Pierre Tignol. On the quaternion ℓ-isogeny path problem. *LMS Journal of Computation and Mathematics*, 17(A):418–432, 2014.

[KMO$^+$] Markulf Kohlweiss, Ueli Maurer, Cristina Onete, Björn Tackmann, and Daniele Venturi. Anonymity-preserving public-key encryption: A constructive approach. In *PETS '13*.

[Koh96] David R. Kohel. *Endomorphism rings of elliptic curves over finite fields*. PhD thesis, University of California, Berkley, 1996.

[Kra10] Hugo Krawczyk. Cryptographic extraction and key derivation: The HKDF scheme. In *Advances in Cryptology – CRYPTO 2010*, pages 631–648. Springer Berlin Heidelberg, 2010.

[Kup05] Greg Kuperberg. A subexponential-time quantum algorithm for the dihedral hidden subgroup problem. *SIAM Journal on Computing*, 35(1):170–188, 2005.

[KYN$^+$21] Yuta Kambe, Masaya Yasuda, Masayuki Noro, Kazuhiro Yokoyama, Yusuke Aikawa, Katsuyuki Takashima, and Momonari Kudo. Solving the constructive deuring correspondence via the kohel-lauter-petit-tignol algorithm. *Mathematical Cryptology*, 1(2):10–24, 2021.

[Lam79] Leslie Lamport. Constructing digital signatures from a one-way function. Technical report, Citeseer, 1979.

[Leh64] Derrick H Lehmer. On a problem of Störmer. *Illinois Journal of Mathematics*, 8(1):57–79, 1964.

[Ler22] Antonin Leroux. *Quaternion Algebra and isogeny-based cryptography*. PhD thesis, Ecole doctorale de l'Institut Polytechnique de Paris, 2022.

[LN11] Florian Luca and Filip Najman. On the largest prime factor of x^2-1. *Mathematics of computation*, 80(273):429–435, 2011.

[Lon22] Patrick Longa. Efficient algorithms for large prime characteristic fields and their application to bilinear pairings and supersingular isogeny-based protocols. *IACR Cryptol. ePrint Arch.*, 2022:367, 2022.

[LP17] David A Levin and Yuval Peres. *Markov chains and mixing times*, volume 107. American Mathematical Soc., 2017.

[LR22] Antonin Leroux and Maxime Roméas. Updatable encryption from group actions. Cryptology ePrint Archive, Paper 2022/739, 2022. `https://eprint.iacr.org/2022/739`.

[LS10] Eyal Lubetzky and Allan Sly. Cutoff phenomena for random walks on random regular graphs. *Duke Mathematical Journal*, 153(3):475–510, 2010.

[Mar99] Greg Martin. An asymptotic formula for the number of smooth values of a polynomial. *Journal of Number Theory*, 93:108–182, 1999.

[McE78] Robert J McEliece. A public-key cryptosystem based on algebraic. *Coding Thv*, 4244:114–116, 1978.

[Mes86] Jean-Francois Mestre. La méthode des graphes. exemples et applications. In *Proceedings of the international conference on class numbers and fundamental units of algebraic number fields (Katata)*, pages 217–242, 1986.

[MMP22] Marzio Mula, Nadir Murru, and Federico Pintore. On random sampling of supersingular elliptic curves. Cryptology ePrint Archive, Paper 2022/528, 2022. `https://eprint.iacr.org/2022/528`.

[MMP$^+$23] Luciano Maino, Chloe Martindale, Lorenz Panny, Giacomo Pope, and Benjamin Wesolowski. A direct key recovery attack on sidh. In Carmit Hazay and Martijn Stam, editors, *Advances in Cryptology – EUROCRYPT 2023*, pages 448–471, Cham, 2023. Springer Nature Switzerland.

[Mon87] Peter L Montgomery. Speeding the pollard and elliptic curve methods of factorization. *Mathematics of computation*, 48(177):243–264, 1987.

[MOT20] Tomoki Moriya, Hiroshi Onuki, and Tsuyoshi Takagi. SiGamal: a supersingular isogeny-based PKE and its application to a PRF. In *Advances in Cryptology–ASIACRYPT 2020: 26th International Conference on the Theory and Application of Cryptology and Information Security, Daejeon, South Korea, December 7–11, 2020, Proceedings, Part II 26*, pages 551–580. Springer, 2020.

[Mur03] M Ram Murty. Ramanujan graphs. *Journal-Ramanujan Mathematical Society*, 18(1):33–52, 2003.

[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, *Advances in Cryptology – ASIACRYPT 2019*, pages 243–272, Cham, 2019. Springer International Publishing.

[NTWZ19] Khoa Nguyen, Hanh Tang, Huaxiong Wang, and Neng Zeng. New code-based privacy-preserving cryptographic constructions. In Steven D. Galbraith and Shiho Moriai, editors, *Advances in Cryptology – ASIACRYPT 2019*, pages 25–55, Cham, 2019. Springer International Publishing.

[PBB13] Albrecht Petzoldt, Stanislav Bulygin, and Johannes Buchmann. A multivariate based threshold ring signature scheme. *Applicable Algebra in Engineering, Communication and Computing*, 24(3-4):255–275, 2013.

[Ped92] Torben Pryds Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In Joan Feigenbaum, editor, *Advances in Cryptology — CRYPTO '91*, pages 129–140, Berlin, Heidelberg, 1992. Springer Berlin Heidelberg.

[Pei20] Chris Peikert. He gives c-sieves on the csidh. In Anne Canteaut and Yuval Ishai, editors, *Advances in Cryptology – EUROCRYPT 2020*, pages 463–492, Cham, 2020. Springer International Publishing.

[Pet17] Christophe Petit. Faster algorithms for isogeny problems using torsion point images. In Tsuyoshi Takagi and Thomas Peyrin, editors, *Advances in Cryptology – ASIACRYPT 2017*, pages 330–353, Cham, 2017. Springer International Publishing.

[Piz80] Arnold Pizer. An algorithm for computing modular forms on $\gamma 0$ (n). *Journal of algebra*, 64(2):340–390, 1980.

[Piz90] Arnold K Pizer. Ramanujan graphs and hecke operators. *Bulletin of the American Mathematical Society*, 23(1):127–137, 1990.

[Piz95] Arnold K Pizer. Ramanujan graphs. computational perspectives on number theory (chicago, il). *AMS/IP Stud. Adv. Math*, 7:159–178, 1995.

[PL17] Christophe Petit and Kristin Lauter. Hard and easy problems for supersingular isogeny graphs. Cryptology ePrint Archive, Report 2017/962, 2017. `https://eprint.iacr.org/2017/962`.

[PLQ08] Christophe Petit, Kristin Lauter, and Jean-Jacques Quisquater. Full cryptanalysis of lps and morgenstern hash functions. In *International Conference on Security and Cryptography for Networks*, pages 263–277. Springer, 2008.

[PR18] Bertram Poettering and Paul Rösler. Towards bidirectional ratcheted key exchange. In *CRYPTO*, 2018.

[PS08] Kenneth G. Paterson and Sriramkrishnan Srinivasan. Security and anonymity of identity-based encryption with multiple trusted authorities. In *Pairing*, 2008.

[PS09] Kenneth G. Paterson and Sriramkrishnan Srinivasan. Building key-private public-key encryption schemes. In *Information Security and Privacy*, 2009.

[Ray18] Dimitrij Ray. *Constructing the Deuring correspondence with applications to supersingular isogeny-based cryptography*. PhD thesis, Master Thesis, Eindhoven University of Technology, 2018.

[Res21] Microsoft Research. Twin smooth integers. `https://github.com/microsoft/twin-smooth-integers`, 2021.

[Rob22] Damien Robert. Some applications of higher dimensional isogenies to elliptic curves (overview of results). Cryptology ePrint Archive, Paper 2022/1704, 2022. `https://eprint.iacr.org/2022/1704`.

[Rob23] Damien Robert. Breaking sidh in polynomial time. In Carmit Hazay and Martijn Stam, editors, *Advances in Cryptology – EUROCRYPT 2023*, pages 472–503, Cham, 2023. Springer Nature Switzerland.

[RS06] Alexander Rostovtsev and Anton Stolbunov. Public-key cryptosystem based on isogenies. Cryptology ePrint Archive, Report 2006/145, 2006. `https://eprint.iacr.org/2006/145`.

[RSA78] R. L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM*, 21(2):120–126, February 1978.

[Sch85] Rene Schoof. Elliptic curves over finite fields and the computation of square roots mod p. *Mathematics of computation*, 44(170):483–494, 1985.

[Sch87] René Schoof. Nonsingular plane cubic curves over finite fields. *Journal of combinatorial theory, Series A*, 46(2):183–211, 1987.

[Sho94] P. W. Shor. Algorithms for quantum computation: discrete logarithms and factoring. In *Proceedings 35th Annual Symposium on Foundations of Computer Science*, pages 124–134, 1994.

[Sho01] Victor Shoup. A proposal for an iso standard for public key encryption. *Cryptology ePrint Archive*, 2001.

[Sho09] Victor Shoup. *A Computational Introduction to Number Theory and Algebra*. Cambridge University Press, USA, 2 edition, 2009.

[Shu01] Chen Shuwen. The prouhet-tarry-escott problem, 2001.

[Sil09] Joseph H Silverman. *The Arithmetic of Elliptic Curves, 2nd Edition*. Graduate Texts in Mathematics. Springer, 2009.

[Sma15] Nigel P. Smart. *Cryptography Made Simple*. Springer Publishing Company, Incorporated, 1st edition, 2015.

[ST15] Ian Stewart and David Tall. *Algebraic number theory and Fermat's last theorem*. CRC Press, 2015.

[Ste21] Bruno Sterner. Commitment schemes from supersingular elliptic curve isogeny graphs. *Mathematical Cryptology*, 1(2):40–51, 2021.

[Stø97] Fredrik C.M. Størmer. Quelques théorèmes sur l'équation de Pell $x^2 - dy^2 = \pm 1$ et leurs applications. *Christiania Videnskabens Selskabs Skrifter, Math. Nat. Kl*, (2):48, 1897.

[Sto09] Anton Stolbunov. Reductionist security arguments for public-key cryptographic schemes based on group action. *Norsk informasjonssikkerhetskonferanse (NISK)*, pages 97–109, 2009.

[Str76] Volker Strassen. Einige resultate über berechnungskomplexität. *Jahresbericht der Deutschen Mathematiker-Vereinigung*, 78:1–8, 1976.

[Sut13] Andrew Sutherland. Isogeny volcanoes. *The Open Book Series*, 1(1):507–530, 2013.

[Sys22] Open Whisper Systems. Signal protocol library for java/android. `https://github.com/signalapp/libsignal-protocol-java`, Accessed 2022.

[Tat66] John Tate. Endomorphisms of abelian varieties over finite fields. *Inventiones mathematicae*, 2(2):134–144, 1966.

[Ten06] Gérald Tenenbaum. Integers with a large friable component. *Acta arithmetica*, 124:287–291, 2006.

[Ten15] Gérald Tenenbaum. *Introduction to analytic and probabilistic number theory*, volume 163. American Mathematical Soc., 2015.

[The16] The National Institute of Standards and Technology (NIST). Submission requirements and evaluation criteria for the post-quantum cryptography standardization process, December, 2016.

[The22] The National Institute of Standards and Technology (NIST). Call for additional digital signature schemes for the post-quantum cryptography standardization process, October, 2022.

[UJ20] David Urbanik and David Jao. New techniques for sidh-based nike. *Journal of Mathematical Cryptology*, 14(1):120–128, 2020.

[Unr15] Dominique Unruh. Non-interactive zero-knowledge proofs in the quantum random oracle model. In *Advances in Cryptology-EUROCRYPT 2015: 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part II 34*, pages 755–784. Springer, 2015.

[Vél71] Jacques Vélu. Isogénies entre courbes elliptiques. *CR Acad. Sci. Paris, Séries A*, 273:305–347, 1971.

[Voi21] John Voight. *Quaternion algebras*. Springer Nature, 2021.

[VOW99] Paul C Van Oorschot and Michael J Wiener. Parallel collision search with cryptanalytic applications. *Journal of cryptology*, 12(1):1–28, 1999.

[Was08] Lawrence C Washington. *Elliptic curves: number theory and cryptography*. CRC press, 2008.

[Wat69] William C Waterhouse. Abelian varieties over finite fields. In *Annales scientifiques de l'École normale supérieure*, volume 2, pages 521–560, 1969.

[Wes22] Benjamin Wesolowski. The supersingular isogeny path and endomorphism ring problems are equivalent. In *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1100–1111. IEEE, 2022.

[XXW13] Xiang Xie, Rui Xue, and Minqian Wang. Zero knowledge proofs from ring-lwe. In Michel Abdalla, Cristina Nita-Rotaru, and Ricardo Dahab, editors, *Cryptology and Network Security*, pages 57–73, Cham, 2013. Springer International Publishing.

[YAJ$^+$17] Youngho Yoo, Reza Azarderakhsh, Amir Jalali, David Jao, and Vladimir Soukharev. A post-quantum digital signature scheme based on supersingular isogenies. In *Financial Cryptography and Data Security: 21st International Conference, FC 2017, Sliema, Malta, April 3-7, 2017, Revised Selected Papers 21*, pages 163–181. Springer, 2017.