A Post-Quantum Digital Signature Scheme based on Supersingular Isogenies

by

Youngho Yoo

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Mathematics
in
Combinatorics and Optimization

Waterloo, Ontario, Canada, 2017

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

We present the first general-purpose digital signature scheme based on supersingular elliptic curve isogenies secure against quantum adversaries in the quantum random oracle model with small key sizes. This scheme is an application of Unruh's construction of non-interactive zero-knowledge proofs to an interactive zero-knowledge proof proposed by De Feo, Jao, and Plût. We implement our proposed scheme on an x86-64 PC platform as well as an ARM-powered device. We exploit the state-of-the-art techniques to speed up the computations for general C and assembly. Finally, we provide timing results for real world applications.

Acknowledgements

I would like to thank my supervisor David Jao.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

The security of public-key cryptosystems depends on the intractability of certain mathematical problems such as integer factorization and computing discrete logarithms. Currently, the most widely-used cryptosystems are based on these two problems which remain infeasible for classical computers to solve. However, there exist efficient quantum algorithms [27] for solving both of these problems, and the realization of large-scale quantum computers pose a serious threat to modern cryptography. Recent efforts to standardize quantum-resistant cryptosystems by the NSA and NIST suggest that the threat of quantum computers may be more real than previously thought.

Post-quantum cryptography is the study of classical cryptosystems that remain secure against quantum adversaries. There are several candidate approaches for building post-quantum cryptographic primitives: lattice-based, code-based, hash-based, and multivariate cryptography. Recently, cryptosystems based on supersingular elliptic curve isogenies were proposed by De Feo, Jao, and Plût [15], who gave protocols for key exchange, zero-knowledge proof of identity, and public key encryption. With small key sizes and efficient implementations [10, 23], isogenies provide a promising candidate for post-quantum key establishment.

Various isogeny-based authentication schemes have been proposed as well, such as strong designated verifier signatures [30], undeniable signatures [21], and undeniable blind signatures [26]. However, it was not known whether isogeny-based cryptography could support general authentication. In this thesis, we show that this is indeed possible by constructing the first general-purpose digital signature scheme based on isogenies which is strongly unforgeable under chosen message attack in the quantum random oracle model.

Our signature scheme is obtained by applying a generic transformation to the zero-knowledge proof of identity proposed in [15]. Classically, obtaining a secure digital signature from an interactive zero-knowledge proof can be achieved by applying the Fiat-Shamir transform [16]. However, its classical security proof requires certain techniques such as rewinding and reprogramming the random oracle which do not necessarily apply in the quantum setting. Quantum rewinding is possible in some restricted cases [33, 36], but it has been shown to be insecure in general [1]. Further, since random oracles model hash functions which, in a real world implementation, could be evaluated in superposition by a quantum adversary, we require *quantum* random oracles which can be queried in superposition. Quantum random oracles are more difficult to reprogram since queries can be in a superposition of exponentially many states and it is difficult to even determine a query input, as measurement disturbs the state.

Unruh [34] recently proposed a transformation which remedies these problems to produce a secure signature from a zero-knowledge proof in the quantum random oracle model. Its overhead is generally much larger than Fiat-Shamir – in some cases exponentially large, making the scheme impractical. Fortunately, applying it to the isogeny-based zero-knowledge proof incurs only twice as much computation as the Fiat-Shamir transform, producing a workable quantum-safe digital signature scheme with small key sizes.

Related Work

Independently of us, Galbraith, Petit, and Silva recently published a preprint containing two isogeny-based digital signature schemes [18]. Their second scheme, based on endomorphism rings, is completely unrelated to our work. Their first scheme, based on the De Feo, Jao, and Plût identification scheme, is conceptually identical to our scheme, but they present significant space optimizations to reduce the signature size down to $12\lambda^2$ bits (or $6\lambda^2$ if non-repudiation is not required), compared to our signature size of $69\lambda^2$ bits. However, we note that their signature size is for classical security level λ and as of this writing their posted preprint contains no signature sizes for post-quantum security, whereas our signature sizes are given in terms of post-quantum security. Moreover, their scheme may be slower, since they use a time-space tradeoff to achieve such small signature sizes. The performance of their scheme is not immediately clear, since they provide no implementation results. In this thesis, by contrast, we provide a complete implementation of our scheme, as well as performance results on multiple platforms and source code for reference.

Outline.

In Chapter 2, we give the necessary mathematical background on isogenies for constructing isogeny-based cryptosystems. In Chapter 3, we give an overview of isogeny-based cryptography and describe the interactive zero-knowledge proof which will be used to construct our signature scheme. In Chapter 4, we describe Unruh's construction. In Chapter 5, we construct and analyze our isogeny-based digital signature scheme, discussing algorithmic aspects, parameter sizes, security, and implementation.

Chapter 2

Background on Isogenies

In this chapter we give a condensed introduction to supersingular isogenies with the goal of describing the necessary background required for understanding our isogeny-based cryptosystems. The theory of elliptic curves and isogenies is deeply rooted in algebraic geometry and it is difficult to give a reasonably brief introduction in full generality. As such, we focus on special cases that are relevant to isogeny-based cryptosystems and omit details and proofs. A full treatment of this subject can be found in [8, 17, 28], which are the main sources for the material summarized here.

2.1 Elliptic Curves

Let $\mathbb{F}$ be a finite field with characteristic p where $p > 3$.

Definition 2.1.1. *An **elliptic curve** E over $\mathbb{F}$, denoted $E/\mathbb{F}$, is a non-singular plane curve satisfying the short Weierstrass equation*

$$y^2 = x^3 + ax + b$$

for some fixed $a, b \in \mathbb{F}$.

The non-singularity condition is equivalent to the statement that the **discriminant** $\Delta := 4a^3 + 27b^2$ is non-zero. In projective coordinates, the elliptic curve equation becomes

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

For a point P on a curve, we use (x_P, y_P) or $(X_P : Y_P : Z_P)$ to denote its affine or projective coordinate elements, respectively. Note that the point $(0 : 1 : 0)$ always satisfies the projective curve equation. This point, denoted $\mathcal{O}$, is called the **point at infinity**. Since there is no corresponding affine point to $\mathcal{O}$, it is usually defined as an extra special point in affine elliptic curves.

Projective curves (or affine curves with the extra point at infinity) admit a well-known group structure with an operation (usually denoted as addition) which can be informally described as follows. Given two points $P, Q \in E$, consider the line passing through P and Q (if $P = Q$, take the tangent to the curve at P). Since elliptic curves are defined by cubic equations, this line meets E at a third point, say R. Then the operation is defined as $P + Q = -R$, where $-R$ denotes the reflection of R across the x-axis (i.e. if $R = (x, y) \equiv (X : Y : Z)$, then $-R = (x, -y) \equiv (X : -Y : Z)$). This operation defines a group structure on elliptic curves with $\mathcal{O}$ as the identity element.

2.2 Isogenies

Definition 2.2.1. *A **rational map** is a map $\phi : E_1 \to E_2$ between elliptic curves E_1, E_2 given by $\phi(P) = (\phi_x(x_P, y_P), \phi_y(x_P, y_P))$, where ϕ_x, ϕ_y are rational functions in x_P, y_P.*

*An **isogeny** is a surjective rational map $\phi : E_1 \to E_2$ that is also a group homomorphism. Two curves are **isogenous** if there exists an isogeny from one to the other.*

Since isogenies are group homomorphisms, they can be identified with their kernels. In implementations, we generally represent isogenies by specifying the generators for its kernel. Conversely, we can compute an isogeny with a given subgroup as the kernel using Vélu's formulas [35], described in §2.4.

Example 2.2.2. *The **multiplication-by-ℓ-map** takes a point P to its scalar multiple $[\ell]P = P + \cdots + P$. It is an isogeny whose kernel is the set of ℓ-torsion points:*

$$E[\ell] := \{P \in E : [\ell]P = \mathcal{O}\}$$

*$E[\ell]$ is the **ℓ-torsion subgroup** of E.*

Torsion subgroups of elliptic curves have a special structure:

Theorem 2.2.3. *Let E be an elliptic curve over $\mathbb{F}$ with characteristic $p > 0$, and let $\ell > 0$ be coprime to p. Then $E[\ell] \cong (\mathbb{Z}/\ell\mathbb{Z})^2$.*

Note that the multiplication-by-ℓ-map is an isogeny that maps a curve to itself. Such isogenies are called **endomorphisms**. For a given curve E, the set of endomorphisms of E forms a ring under pointwise addition and composition called the **endomorphism ring** of E, denoted $End(E)$.

For an elliptic curve $E : y^2 = x^3 + ax + b$ over $\mathbb{F}$, its **coordinate ring** is

$$\mathbb{F}[E] := \mathbb{F}[x, y]/\langle y^2 - x^3 - ax - b\rangle$$

where $\mathbb{F}[x, y]$ is the ring of polynomials over $\mathbb{F}$. Its **function field**, denoted $\mathbb{F}(E)$, is the field of fractions of $\mathbb{F}[E]$.

For a given isogeny $\phi : E_1 \to E_2$, we define its **pullback**, denoted ϕ^*, to be the map $\phi^* : \mathbb{F}(E_2) \to \mathbb{F}(E_1)$ where for $f \in \mathbb{F}(E_2)$, $\phi^*(f) = f \circ \phi$. An isogeny $\phi : E_1 \to E_2$ is **separable** if the field extension $\mathbb{F}(E_1)/\phi^*(\mathbb{F}(E_2))$ is separable (i.e. each element of $\mathbb{F}(E_2)$ has a separable minimal polynomial over $\phi^*(\mathbb{F}(E_2))$).

The **degree** of the isogeny is the degree of the field extension $[\mathbb{F}(E_1) : \phi^*(\mathbb{F}(E_2))]$.

Proposition 2.2.4. *For a separable isogeny, its degree is equal to the cardinality of its kernel.*

Our isogeny-based cryptosystems only use separable isogenies, and we will assume that all isogenies are separable for the remainder of this thesis.

Proposition 2.2.5. *Every isogeny $\phi : E_1 \to E_2$ has a unique* ***dual isogeny*** *$\hat{\phi} : E_2 \to E_1$ with the same degree such that $\hat{\phi} \circ \phi$ and $\phi \circ \hat{\phi}$ are the multiplication maps by $deg(\phi)$ on E_1 and E_2 respectively.*

Thus being isogenous defines an equivalence relation. Tate's theorem [32] shows that the isogeny classes of curves over finite fields are characterized by their cardinality:

Theorem 2.2.6. *Two curves over $\mathbb{F}$ are isogenous over $\mathbb{F}$ (i.e. the isogeny is defined over $\mathbb{F}$) if and only if they have the same cardinality.*

An **isomorphism** between elliptic curves is an isogeny with degree 1. Isomorphism classes of elliptic curves are characterized by their ***j*-invariants** which is defined for a curve $E : y^2 = x^3 + ax + b$ as follows:

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

In other words, two elliptic curves are isomorphic (over the closure of $\mathbb{F}$) if and only if they have the same j-invariant.

2.3 Supersingularity

The endomorphism rings of elliptic curves over finite fields are usually isomorphic to an order in an imaginary quadratic field. However, some curves have much larger endomorphism rings isomorphic to an order in a quaternion algebra.

Definition 2.3.1. *An elliptic curve E is **supersingular** if its endomorphism ring is isomorphic to an order in a quaternion algebra. Otherwise it is **ordinary** and its endomorphism ring is isomorphic to an imaginary quadratic field.*

In particular, endomorphism rings of ordinary curves are commutative, while supersingular endomorphism rings are not. Childs, Jao, and Soukharev [7] gave a quantum subexponential-time algorithm for computing ordinary isogenies by a reduction to a hidden shift problem. However, their technique does not apply in the supersingular case due to the non-commutativity, and the best known algorithms for computing supersingular isogenies remain fully exponential time.

A supersingular curve cannot be isogenous to an ordinary curve. A **supersingular (resp. ordinary) isogeny** is an isogeny between supersingular (resp. ordinary) curves.

Although there exist supersingular curves over $\mathbb{F}_{p^e}$ for all $e \geq 1$, all supersingular curves are isomorphic to curves defined over $\mathbb{F}_{p^2}$.

2.4 Vélu's Formulas

Vélu's formulas [35] explicitly describe how to compute an isogeny with a given kernel. Let $E : y^2 = x^3 + ax + b$ be an elliptic curve and let G be a finite subgroup of E. Let G_2 be the subset of points of G with order 2, and let G_1 be such that

$$G = \{\mathcal{O}\} \;\dot\cup\; G_2 \;\dot\cup\; G_1 \;\dot\cup\; (-G_1)$$

where $-G_1 = \{-P : P \in G_1\}$ and $\dot\cup$ denotes disjoint union.

For $Q = (x_Q, y_Q) \in G_1 \cup G_2$, define

$$\begin{aligned} F_x(Q) &= 3x_Q^2 + a \\ F_y(Q) &= -2y_Q \\ u(Q) &= (F_y(Q))^2 \\ t(Q) &= \begin{cases} F_x(Q) & \text{if } Q \in G_2 \\ 2F_x(Q) & \text{if } Q \in G_1 \end{cases} \end{aligned}$$

and

$$\begin{aligned} A &= a - 5 \cdot \sum_{Q \in G_1 \cup G_2} t(Q) \\ B &= b - 7 \cdot \sum_{Q \in G_1 \cup G_2} (u(Q) + x_Q t(Q)) \end{aligned}$$

Vèlu showed that the isogeny ϕ on E with kernel G is defined by the map $\phi : (x, y) \mapsto (X, Y)$ where

$$\begin{aligned} X &= x + \sum_{Q \in G_1 \cup G_2} \frac{t(Q)}{x - x_Q} + \frac{u(Q)}{(x - x_Q)^2} \\ Y &= y - \sum_{Q \in G_1 \cup G_2} \frac{u(Q) \cdot 2y}{(x - x_Q)^3} + \frac{t(Q)(y - y_Q) - F_x(Q)F_y(Q)}{(x - x_Q)^2} \end{aligned}$$

and the image curve of ϕ is given by

$$E' : y^2 = x^3 + Ax + B$$

Since the computation involves a sum over each element in $G_1 \cup G_2$, its running time is proportional to the size of the kernel, which may be exponentially large. However, as we will see in §5.2.3, the isogeny can be decomposed and computed efficiently when the size of the kernel is a power of a small prime.

Chapter 3

Isogeny-Based Cryptography

3.1 Background

Isogeny-based cryptosystems rely on the difficulty of computing the isogeny between two given curves of the same order. The first isogeny-based cryptosystems were proposed by Couveignes [12] and Rostovtsev and Stolbunov [25] in 2006, where ordinary isogenies were used to construct schemes for public-key encryption and key exchange. However, in 2010, Childs, Jao, and Soukharev [7] gave a quantum algorithm which, under the Generalized Riemann Hypothesis, can compute ordinary isogenies in subexponential time. This raised serious concerns on the practicality of cryptosystems based on ordinary isogenies, given their already poor performance (229 seconds to perform a key exchange operation for 128-bit security under previous assumptions [29]).

Since the algorithm relied on the commutativity of the endomorphism rings, it did not apply to the supersingular case, and the best known algorithms for computing supersingular isogenies remained exponential time. In 2011, Jao and De Feo [20] successfully constructed cryptosystems based on supersingular isogenies for encryption and key exchange (now known as the Supersingular Isogeny Diffie Hellman, or SIDH), and gave an implementation achieving much faster performance compared to those based on ordinary isogenies. This paper was later extended with Plût [15] to include a scheme for zero-knowledge proof of identity and further optimizations to the implementation, achieving a runtime of roughly 0.06 seconds per key exchange operation.

In 2016, Costello, Longa, and Naehrig [10] published a constant-time implementation of SIDH with more efficient algorithms for computing isogenies, running up to 2.9 times faster than the implementation of Jao, De Feo, and Plût [15]. Techniques for compressing public keys of isogeny-based cryptosystems (which were already quite small) were proposed in [2] and optimized in [9].

There are several specialized authentication schemes based on supersingular isogenies [30, 21, 26], but a general-purpose digital signature scheme had not yet been proposed. This thesis (and independently the work of Galbraith, Petit, and Silva [18]) addresses this problem by constructing an isogeny-based digital signature scheme, and we also provide implementation results in §5.2.

3.2 Protocols

We describe the Supersingular Isogeny Diffie-Hellman (SIDH) key exchange and the zero-knowledge proof of identity protocols proposed by Jao et al. [15]. Although we do not directly need SIDH in our signature scheme, we make use of results on efficient implementations of isogeny-based systems which focus on SIDH, and it would feel incomplete to leave it out.

We give a mainly theoretical description in this chapter, and leave all computational and algorithmic details to Chapter 5.

3.2.1 Setup

Isogeny-based cryptosystems use supersingular elliptic curves over fields of characteristic p, where the prime p has the form $p = \ell_A^{e_A} \ell_B^{e_B} f \pm 1$ such that ℓ_A, ℓ_B are small primes (typically 2 and 3) with $\ell_A^{e_A} \approx \ell_B^{e_B}$, and f is a small cofactor to ensure p is prime. This special form of p allows us to efficiently compute isogenies, as we will see in §5.2.3.

The public parameters consist of the prime $p = \ell_A^{e_A} \ell_B^{e_B} f \pm 1$, a supersingular elliptic curve E over $\mathbb{F}_{p^2}$ of order $(\ell_A^{e_A} \ell_B^{e_B} f)^2$, and generators (P_A, Q_A) and (P_B, Q_B) of the $\ell_A^{e_A}$ and $\ell_B^{e_B}$-torsion subgroups $E[\ell_A^{e_A}]$ and $E[\ell_B^{e_B}]$ respectively.

The isogeny-based cryptosystems we will describe are based on the commutativity of the diagram in Figure 3.1.

In Figure 3.1, we have the starting supersingular curve E and torsion points A and B of order $\ell_A^{e_A}$ and $\ell_B^{e_B}$, respectively, on E. The subgroups generated by A and B uniquely correspond to isogenies ϕ_A and ϕ_B with kernel $\langle A \rangle$ and $\langle B \rangle$, respectively. The image curves of ϕ_A and ϕ_B are denoted $E/\langle A \rangle$ and $E/\langle B \rangle$, respectively.

$$\begin{tikzcd}
E \arrow[r, "\phi_A"] \arrow[d, "\phi_B"'] & E/\langle A\rangle \arrow[d, "\phi'_B"] \\
E/\langle B\rangle \arrow[r, "\phi'_A"] & E/\langle A, B\rangle
\end{tikzcd}$$

Figure 3.1: Each arrow is labelled by the isogeny and its kernel.

The torsion point A has image $\phi_B(A)$ in the curve $E/\langle B\rangle$, with order $\ell_A^{e_A}$. The torsion subgroup $\langle \phi_B(A)\rangle$ corresponds to an isogeny ϕ'_A from $E/\langle B\rangle$ to $(E/\langle B\rangle)/\langle \phi_B(A)\rangle = E/\langle A, B\rangle$. The isogeny $\phi'_B : E/\langle A\rangle \to E/\langle A, B\rangle$ is similarly defined, with kernel $\langle \phi_A(B)\rangle$.

3.2.2 Key Exchange

Suppose Alice and Bob want to establish a shared secret key. Alice computes random linear combinations of P_A, Q_A to obtain a random full-order $\ell_A^{e_A}$-torsion point $A = [m_A]P_A + [n_A]Q_A$, and computes the isogeny ϕ_A with kernel $\langle A\rangle$. This isogeny, represented by the point A, is Alice's private key. For the public key, Alice publishes the image curve $E/\langle A\rangle$, along with the images of the public generators (P_B, Q_B) under her isogeny ϕ_A. Bob similarly chooses a random $\ell_B^{e_B}$-torsion point B, computes ϕ_B, and publishes $E/\langle B\rangle, \phi_B(P_A), \phi_B(Q_A)$.

To compute the shared secret, Alice needs to compute the point $\phi_B(A)$ which generates the kernel of the isogeny $\phi'_A : E/\langle B\rangle \to E/\langle A, B\rangle$. Since Bob has published $(\phi_B(P_A), \phi_B(Q_A))$ and isogenies are group homomorphisms, she can compute

$$\phi_B(A) = \phi_B([m_A]P_A + [n_A]Q_A) = [m_A]\phi_B(P_A) + [n_A]\phi_B(Q_A),$$

allowing her to obtain the image curve $E/\langle A, B\rangle$ of the isogeny ϕ'_A. Similarly, Bob computes $\phi_A(B) = [m_B]\phi_A(P_B) + [n_B]\phi_A(Q_B)$ and the isogeny $\phi'_B : E/\langle A\rangle \to E/\langle A, B\rangle$. Alice and Bob can then take the j-invariant of $E/\langle A, B\rangle$ to arrive at a shared secret.

3.2.3 Zero-Knowledge Proof of Identity

The zero-knowledge proof works over the same diagram, but for clarity we will use different labels as shown in Figure 3.2.3.

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

Peggy (the prover) has a secret point S generating the kernel of the isogeny $\phi\colon E \to E/\langle S\rangle$. Her private key is S (or any generator of $\langle S\rangle$) and her public key is the curve $E/\langle S\rangle$ and the images of the public generators $\phi(P_B), \phi(Q_B)$.

In order to prove her identity (i.e. her knowledge of $\langle S\rangle$) to Vic (the verifier), Peggy chooses a random point R of order $\ell_B^{e_B}$ defining an isogeny $\psi\colon E \to E/\langle R\rangle$. She then computes and sends the curves $E/\langle R\rangle$ and $E/\langle R, S\rangle$ to Vic.

At this point, Vic knows all four curves in the diagram, but none of the isogenies between them. To verify, Vic challenges Peggy to reveal some of the isogenies by sending her a random challenge bit $b \in \{0, 1\}$. Depending on b, Peggy reveals either (ψ, ψ') or ϕ', as shown in Figure 3.2.

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

$b = 0$

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

$b = 1$

Figure 3.2: Hidden isogenies are indicated by dashed lines. Bolded lines indicate the isogenies revealed by Peggy on challenge b. In either case, the revealed isogenies do not leak information about the secret isogeny ϕ.

More precisely, Peggy and Vic run the following protocol:

1. - Peggy chooses a random point R of order $\ell_B^{e_B}$.

- She computes the isogeny $\psi\colon E \to E/\langle R\rangle$.
- She computes the isogeny $\phi'\colon E/\langle R\rangle \to E/\langle R, S\rangle$ with kernel $\langle\psi(S)\rangle$ (alternatively the isogeny $\psi'\colon E/\langle S\rangle \to E/\langle R, S\rangle$ with kernel $\langle\phi(R)\rangle$)
- She sends the commitment `com` $= (E_1, E_2)$ to Vic, where $E_1 = E/\langle R\rangle$ and $E_2 = E/\langle R, S\rangle$.

2. Vic randomly chooses a challenge bit `ch` $\in \{0, 1\}$ and sends it to Peggy.

3. Peggy sends the response `resp` where

   - If `ch` $= 0$, then `resp` $= (R, \phi(R))$.
   - If `ch` $= 1$, then `resp` $= \psi(S)$.

4. - If `ch` $= 0$, Vic verifies that R and $\phi(R)$ have order $\ell_B^{e_B}$ and generate the kernels for the isogenies $E \to E_1$ and $E/\langle S\rangle \to E_2$ respectively.
   - If `ch` $= 1$, Vic verifies that $\psi(S)$ has order $\ell_A^{e_A}$ and generates the kernel for the isogeny $E_1 \to E_2$.

To achieve λ bits of security, the prime p should be roughly 6λ bits and this protocol should be run λ times. If Vic successfully verifies all λ rounds of the protocol, then Peggy has proved her identity (knowledge of the private key S) to Vic. Otherwise, Vic rejects.

3.3 Security

We prove the security of the zero-knowledge proof of identity, but leave out the proof of the key exchange protocol since its security is not directly related to the construction of our signature scheme.

The required security assumptions are based on the following problems from [15, §5], which are believed to be intractable even for quantum computers.

Computational Supersingular Isogeny (CSSI) problem: Let $\phi_A\colon E_0 \to E_A$ be an isogeny whose kernel is $\langle R_A\rangle$ where R_A is a random point with order $\ell_A^{e_A}$. Given $E_A, \phi_A(P_B), \phi_A(Q_B)$, find a generator of $\langle R_A\rangle$.

Decisional Supersingular Product (DSSP) problem: Let $\phi\colon E_0 \to E_3$ be an isogeny of degree $\ell_A^{e_A}$. Given (E_1, E_2, ϕ') sampled with probability $1/2$ from one or the other of the following distributions, determine which distribution it is from.

- A random point R of order $\ell_B^{e_B}$ is chosen and $E_1 = E_0/\langle R\rangle$, $E_2 = E_3/\langle\phi(R)\rangle$, and $\phi'\colon E_1 \to E_2$ is an isogeny of degree $\ell_A^{e_A}$.
- E_1 is chosen randomly among curves of the same cardinality as E_0, and $\phi'\colon E_1 \to E_2$ is a random isogeny of degree $\ell_A^{e_A}$

The CSSI (resp. DSSP) assumption is the assumption that the CSSI (resp. DSSP) problem is computationally infeasible, even for quantum computers.

The best known attack for the CSSI problem involves claw-finding algorithms using quantum walks [31] and takes $O(p^{1/6})$ time, which is optimal for a black-box claw attack [39]. Therefore it is believed that a prime with bitlength 6λ achieves λ bits of post-quantum security.

3.3.1 Security of the Zero-Knowledge Proof

The security properties we will prove are completeness, special soundness, and honest-verifier zero-knowledge (HVZK). The definitions for completeness and HVZK are standard, but special soundness differs slightly from the usual soundness property. It states that, given any two valid responses to the same challenge, one can extract the private key. Special soundness is required for Unruh's construction, and these properties are formally defined in §4.1.

Theorem 3.3.1 (adapted from [15, Theorem 6.3])**.** *The isogeny-based zero-knowledge proof of identity satisfies completeness, special soundness, and honest-verifier zero-knowledge.*

Proof. Completeness follows from the fact that the diagram in Figure 3.2.3 commutes. As long as Peggy follows the protocol, her responses should always be verifiable.

For special soundness, suppose we are given two valid transcripts $(\texttt{com}, 0, \texttt{resp}_0)$ and $(\texttt{com}, 1, \texttt{resp}_1)$, where $\texttt{com} = (E_1, E_2)$. Then we can use $\texttt{resp}_0 = (R, \phi(R))$ to compute the isogeny $\psi\colon E \to E/\langle R\rangle$. Since $\texttt{resp}_1 = \psi(S)$ is a generator of the kernel of ϕ', we can take the dual isogeny $\hat{\psi}\colon E/\langle R\rangle \to E$, and compute $\hat{\psi}(\texttt{resp}_1)$, a generator for $\langle S\rangle$ (see Figure 3.3).

To prove honest-verifier zero-knowledge, we show that we can simulate outputs $(\texttt{com}, \texttt{ch}, \texttt{resp})$ that are indistinguishable from valid interactions between a prover and an honest verifier. To do this, we first choose a random bit $\texttt{ch} \in \{0, 1\}$.

If $\texttt{ch} = 0$, then we follow the protocol by choosing a random $\ell_B^{e_B}$-torsion point R (and $\phi(R)$ using the public parameters) and computing $\texttt{com} = (E/\langle R\rangle, E/\langle R, S\rangle)$. Setting $\texttt{resp} = (\psi, \psi')$, we have a valid interaction $(\texttt{com}, \texttt{ch}, \texttt{resp})$.

If $\texttt{ch} = 1$, we choose a random supersingular curve E_1 and a random $\ell_A^{e_A}$-torsion point S' on E_1, defining an isogeny $\phi' : E_1 \to E_1/\langle S'\rangle$. Then we set $E_2 = E_1/\langle S'\rangle$, $\texttt{com} = (E_1, E_2)$, and $\texttt{resp} = \phi'$. By the DSSP assumption, this output is indistinguishable from a valid interaction. □

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

Figure 3.3: If ψ and ϕ' are both known, then we can recover the secret subgroup $\langle S\rangle$.

3.4 Compression

Azarderakhsh et al. [2] introduced two techniques for compressing parameters in isogeny-based cryptosystems. However their implementation was quite slow compared to the runtime of the isogeny computations. Costello et al. [9] proposed new algorithms for the compression, accelerating the previous work by more than an order of magnitude to achieve a runtime that is roughly as fast as the isogeny computations while reducing the public key sizes slightly further.

The first technique of [2] is simple: we can represent an elliptic curve $E : y^2 = x^3+ax+b$ by its j-invariant $j(E) \in \mathbb{F}_{p^2}$ instead of the two parameters $a, b \in \mathbb{F}_{p^2}$, cutting storage requirements by a half.

The second technique has to do with representing the public torsion bases. To represent a torsion point $P = (x_P, y_P)$ in the straightforward way, we need two field elements $x_P, y_P \in \mathbb{F}_{p^2}$. Since the coordinates satisfy the elliptic curve equation $y^2 = x^3 + ax + b$, there are only two possible values of y_P for a given x_P, so it can suffice to store just one field element $x_P \in \mathbb{F}_{p^2}$ and an additional indicator bit.

The idea of [2] is as follows: just as we can represent a random full-order torsion point by their coefficients with respect to the public basis, we can represent each public torsion basis point in terms of their coefficients with respect to some other fixed basis. The new basis need not be published as a public parameter, as long as all parties are able to generate the same basis independently by a deterministic algorithm. Since the discrete logarithm problem is easy on smooth-order curves, one can easily compute the coefficients of the public torsion basis points with respect to the new deterministically generated basis. With this approach, each torsion basis point can be represented with two smaller coefficients, also reducing the storage requirements by a half (we will see a more detailed breakdown in §5.3). In [9], the public torsion basis is further compressed by using only three coefficients to represent both basis points.

Chapter 4

Unruh's Construction

Unruh's construction [34] is a generic transformation which takes an interactive zero-knowledge proof system and produces a non-interactive one, like the Fiat-Shamir transform [16]. In contrast to the Fiat-Shamir however, Unruh's construction satisfies a property called *online extractability* which allows us to extract the witness (private key) from a successful adversary without rewinding, a technique that is problematic in the quantum setting. It also avoids the problem of determining the query inputs of the quantum random oracle by including its outputs in the proof (signature) and "inverting" them in the security proof.

Let R be a binary relation. We say that a statement x holds if there exists w such that $(x, w) \in R$. In this case, we call w a **witness** to x. In a proof system, a prover P tries to prove a statement x to a verifier V (in other words, P tries to convince V that P knows a witness w to x). We assume that all parties have access to a quantum random oracle H which can be queried in superposition.

4.1 Sigma Protocols

A **sigma protocol** $\Sigma = (P, V)$, where $P = (P^1, P^2)$, is an interactive proof system consisting of three messages in order: a **commitment** $\texttt{com} = P^1(x, w)$ made by the prover, a **challenge** $\texttt{ch}$ chosen uniformly at random by the verifier, and the **response** $\texttt{resp} = P^2(x, w, \texttt{com}, \texttt{ch})$ computed by the prover based on the challenge. Then, based on this interaction, V outputs $V(x, \texttt{com}, \texttt{ch}, \texttt{resp}) \in \{0, 1\}$, indicating whether they accept or reject the proof.

Let $\Sigma = (P, V)$ be a sigma protocol where $P = (P^1, P^2)$. We define the following properties of sigma protocols (from [34, §2.2]):

Completeness: If P knows a witness w to the statement x, then V accepts.

Special soundness: There exists a polynomial time extractor E_Σ such that, given any pair $(\texttt{com}, \texttt{ch}, \texttt{resp})$ and $(\texttt{com}, \texttt{ch}', \texttt{resp}')$ of valid interactions (accepted by V) with $\texttt{ch} \neq \texttt{ch}'$, E_Σ can compute a witness w such that $(x, w) \in R$.

Honest-verifier zero-knowledge (HVZK): There is a polynomial time simulator S_Σ with outputs of the form $(\texttt{com}, \texttt{ch}, \texttt{resp})$ that are indistinguishable from valid interactions between a prover and an honest verifier by any quantum polynomial time algorithm.

Recall that the isogeny-based zero-knowledge proof of identity from §3.2.3 is a sigma protocol satisfying completeness, special soundness, and honest-verifier zero-knowledge.

4.2 Non-interactive Proof Systems

A **non-interactive proof system** consists of two algorithms: a prover $P(x, w)$ outputting a proof π of the statement x with witness w, and a verifier $V(x, \pi)$ outputting whether it accepts or rejects the proof π of x.

For a non-interactive proof system (P, V), we define the following properties (from [34, §2.1]):

Completeness: If $(x, w) \in R$, then V accepts the proof $\pi = P(x, w)$.

Zero-knowledge (NIZK): There exists a polynomial time simulator S such that, given the ability to program the random oracle H, S can output proofs indistinguishable from those produced by P by any quantum polynomial time algorithm.

The simulator is modeled by two algorithms $S = (S_{\texttt{init}}, S_P)$, where $S_{\texttt{init}}$ outputs an initial circuit H simulating a quantum random oracle, and S_P is a stateful algorithm which may reprogram H and produce proofs using H.

Simulation-sound online-extractability: (with respect to a simulator $S = (S_{\texttt{init}}, S_P)$) There exists a polynomial time extractor E such that, if a quantum polynomial-time algorithm $\mathcal{A}$ with quantum access to $H \leftarrow S_{\texttt{init}}$ and classical access to the prover S_P

outputs a new valid proof of a statement x, then E can compute (extract) a witness w of x.

Remark 4.2.1. *Granting $\mathcal{A}$ classical access to the simulated prover S_P is analogous to granting the adversary access to a classical signing oracle in a chosen message attack in the context of signatures. We could allow $\mathcal{A}$ to have* quantum *access to S_P, corresponding to a* quantum *chosen message attack as defined in [6]. We do not know whether Unruh's construction remains secure under this relaxation.*

4.3 Unruh's Construction

Unruh's construction transforms a sigma protocol Σ into a non-interactive proof system (P_{OE}, V_{OE}) so that, if Σ satisfies completeness, special soundness, and HVZK, then the result is a complete NIZK proof system with simulation-sound online-extractability.

Suppose we have a sigma protocol $\Sigma = (P_\Sigma, V_\Sigma)$ with $P_\Sigma = (P_\Sigma^1, P_\Sigma^2)$, where there are c possible challenges in the challenge domain N_{ch} and the parties want to run the protocol t times, where t depends on the security parameter λ (in our signature scheme we will have $N_{ch} = \{0, 1\}$, $c = 2$, and $t = 2\lambda$). Let G, H be quantum random oracles, where G has the same domain and range. We define a non-interactive proof system (P_{OE}, V_{OE}) where P_{OE} and V_{OE} are given by Algorithms 1 and 2 respectively.

The idea is to simulate the interaction in Σ by setting the challenge $J = J_1 \| \dots \| J_t$ as the output of the random function H. However, instead of evaluating H on the commitments $(\texttt{com}_i)_i$ alone as in the Fiat-Shamir transform, we also include the hashes $h_{i,j} = G(\texttt{resp}_{i,j})$ of the responses $\texttt{resp}_{i,j}$ to each possible challenge $\texttt{ch}_{i,j}$, for each commitment $\texttt{com}_i$. Then the produced proof consists of the commitments, an ordering of all possible challenges, hashed responses to the corresponding challenges, and the responses to the challenges given by $J_1 \| \dots \| J_t$. The verifier can then take the data to reproduce $J_1 \| \dots \| J_t$, check that the data was produced properly, and verify the responses $(\texttt{resp}_{i,J_i})_i$ for each round of Σ.

The main theorem of [34] proves that this construction is secure (that it satisfies the three properties defined in the previous section) in the quantum oracle model. The main idea of the proof is that, since the random oracle G has the same domain and range, it is indistinguishable from a random permutation [38], thus it can be replaced by an efficiently invertible function which is indistinguishable from a random oracle (for example, a random polynomial of high enough degree [37]). This allows the hashes in the proof to be inverted

Algorithm 1 Prover: P_{OE} on input (x, w)

```
// Create $t \cdot c$ proofs and hash each response
for $i = 1$ to $t$ do
  $\texttt{com}_i \leftarrow P^1_\Sigma(x, w)$
  for $j = 1$ to $c$ do
    $\texttt{ch}_{i,j} \leftarrow_R N_{ch} \setminus \{\texttt{ch}_{i,1}, \ldots, \texttt{ch}_{i,j-1}\}$
    $\texttt{resp}_{i,j} \leftarrow P^2_\Sigma(x, w, \texttt{com}_i, \texttt{ch}_{i,j})$
    $h_{i,j} \leftarrow G(\texttt{resp}_{i,j})$
// Get challenge by hashing
$J_1 \| \ldots \| J_t \leftarrow H(x, (\texttt{com}_i)_i, (\texttt{ch}_{i,j})_{i,j}, (h_{i,j})_{i,j})$
// Return proof
return $\pi \leftarrow ((\texttt{com}_i)_i, (\texttt{ch}_{i,j})_{i,j}, (h_{i,j})_{i,j}, (\texttt{resp}_{i,J_i})_i)$
```

Algorithm 2 Verifier: V_{OE} on input (x, π), where $\pi = ((\texttt{com}_i)_i, (\texttt{ch}_{i,j})_{i,j}, (h_{i,j})_{i,j}, (\texttt{resp}_{i,J_i})_i)$

```
// Compute the challenge hash
$J_1 \| \ldots \| J_t \leftarrow H(x, (\texttt{com}_i)_i, (\texttt{ch}_{i,j})_{i,j}, (h_{i,j})_{i,j})$
for $i = 1$ to $t$ do
  check $\texttt{ch}_{i,1}, \ldots, \texttt{ch}_{i,c}$ pairwise distinct
  check $h_{i,J_i} = G(\texttt{resp}_i)$
  check $V_\Sigma(x, \texttt{com}_i, \texttt{ch}_{i,J_i}, \texttt{resp}_i) = 1$
if all checks succeed then
  return 1
```

to obtain the hidden responses in the adversary's forged proof, which allows us to obtain the private key by special soundness.

Theorem 4.3.1 ([34, Corollary 19])**.** *If Σ satisfies completeness, special soundness, and HVZK, then (P_{OE}, V_{OE}) is a complete non-interactive zero-knowledge proof system with simulation-sound online extractability in the quantum random oracle model.*

4.4 Signatures from Non-interactive Zero-Knowledge Proofs

Definition 4.4.1. *A* ***digital signature*** *scheme consists of three algorithms:*

- $\texttt{Keygen}(\lambda)$*: takes a security parameter* λ *and outputs a public-private key pair* $(\texttt{pk}, \texttt{sk})$*.*

- $\texttt{Sign}(\texttt{sk}, m)$*: signs the message* m *using the private key* $\texttt{sk}$*, outputting a signature* σ*.*

- $\texttt{Verify}(\texttt{pk}, m, \sigma)$*: takes the public key of the claimed signer and verifies the signature* σ *on the message* m*.*

Definition 4.4.2. *A digital signature scheme is* ***strongly unforgeable under chosen message attack (SUF-CMA)*** *if, for any quantum polynomial time adversary* $\mathcal{A}$ *with classical access to the signing oracle* $\texttt{sig}\colon m \mapsto \texttt{Sign}(\texttt{sk}, m)$*,* $\mathcal{A}$ *cannot produce a new valid message-signature pair with non-negligible probability.*

Suppose we have a function $\texttt{Keygen}$ taking a security parameter λ and generating a public-private key pair $(\texttt{pk}, \texttt{sk})$ such that no quantum polynomial-time algorithm can recover a valid $\texttt{sk}$ from $\texttt{pk}$ with non-negligible probability. We say that such a function is a **hard instance generator**.

Since a proof of identity just proves knowledge of the corresponding $\texttt{sk}$ for a given $\texttt{pk}$, in the context of proof systems it can be viewed as proving the statement $x = \texttt{pk}$ with witness $w = \texttt{sk}$, where $(x, w) \in R$ if and only if (x, w) is a valid key pair that can be generated by a hard instance generator $\texttt{Keygen}$.

From this view, a digital signature is just a non-interactive zero-knowledge proof of identity for $\texttt{Keygen}$, except that this ignores the message that is being signed. To incorporate a specific message into each proof (or signature), we can simply include the message as part of the statement while the relation R simply ignores the message. In other words, the statement being proved is of the form $x = (\texttt{pk}, m)$, and for the relation R, we have $((\texttt{pk}, m), w) \in R$ if and only if $(\texttt{pk}, w)$ is a valid key pair generated by $\texttt{Keygen}$. Thus, from a NIZK proof of identity (P, V), we obtain a digital signature scheme $\mathcal{DS} = (\texttt{Keygen}, \texttt{Sign}, \texttt{Verify})$ where $\texttt{Sign}(\texttt{sk}, m) = P((\texttt{pk}, m), \texttt{sk})$ and $\texttt{Verify}(\texttt{pk}, m, \sigma) = V((\texttt{pk}, m), \sigma)$.

Theorem 4.4.3 ([34, Theorem 23])**.** *Let* $\texttt{Keygen}$ *be a hard instance generator and* (P, V) *a non-interactive proof of identity for* $\texttt{Keygen}$ *satisfying completeness, zero-knowledge, and simulation-sound online-extractability. Define:*

- $\texttt{Sign}(\texttt{sk}, m) = P((\texttt{pk}, m), \texttt{sk})$

- $\texttt{Verify}(\texttt{pk}, m, \sigma) = V((\texttt{pk}, m), \sigma)$*.*

Then the digital signature scheme $\mathcal{DS} = (\mathtt{Keygen}, \mathtt{Sign}, \mathtt{Verify})$ *is strongly unforgeable under chosen message attack (SUF-CMA) in the quantum random oracle model.*

Proof. Suppose there exists a quantum polynomial time adversary A that can forge a new valid message-signature pair, given access to a classical signing oracle. Since (P, V) is zero-knowledge, there is a polynomial time simulator $S = (S_{\mathtt{init}}, S_P)$ that can indistinguishably simulate proofs. Then we can use S to simulate the signing oracle (on a signing query m, output a simulated proof of the statement $(\mathtt{pk}, m)$), and since A cannot distinguish the simulated signatures, A will be able to forge a new valid message-signature pair, say (m, σ). Then σ is a new valid proof of the statement $(\mathtt{pk}, m)$, thus by simulation-sound online-extractability, we can efficiently extract a witness $\mathtt{sk}$ of $\mathtt{pk}$. This contradicts the assumption that $\mathtt{Keygen}$ is a hard instance generator. □

Chapter 5

Isogeny-Based Digital Signature

Let Σ denote the isogeny-based zero-knowledge proof of identity described in §3.2.3. Applying Unruh's construction to Σ, we obtain a non-interactive proof of identity (P_{OE}, V_{OE}), from which we get a digital signature scheme by the method described in §4.4.

5.1 Signature Scheme

Public Parameters. We have the same public parameters as in Σ:

- A prime $p = \ell_A^{e_A}\ell_B^{e_B} f \pm 1$
- A supersingular elliptic curve E of cardinality $(\ell_A^{e_A}\ell_B^{e_B})^2$ over $\mathbb{F}_{p^2}$
- Generators (P_B, Q_B) of the torsion group $E[\ell_B^{e_B}]$

Key Generation. To generate a public-private key pair $(\mathtt{pk}, \mathtt{sk})$:

1. Choose a random point S of order $\ell_A^{e_A}$
2. Compute the isogeny $\phi \colon E \to E/\langle S \rangle$
3. Output $(\mathtt{pk}, \mathtt{sk})$ where $\mathtt{pk} = (E/\langle S\rangle, \phi(P_B), \phi(Q_B))$ and $\mathtt{sk} = S$.

Signing. To sign a message m with the key $\mathtt{sk}$,

1. Run the zero-knowledge proof of identity 2λ times:
   (a) Choose a random point R_i of order $\ell_B^{e_B}$

   (b) Compute the commitment $\texttt{com}_i = (E/\langle R\rangle, E/\langle R, S\rangle)$ by computing the isogenies $\psi : E \to E/\langle R\rangle$ and either $\phi' : E/\langle R\rangle \to E/\langle R, S\rangle$ or $\psi' : E/\langle S\rangle \to E/\langle R, S\rangle$
   (c) Choose a random challenge order $(\texttt{ch}_{i,j})_j$, the responses $\texttt{resp}_{i,j}$ for each challenge, and their hashes $h_{i,j}$
2. Hash all the commitments, challenges, and hashes of responses together with the public key $\texttt{pk}$ and the message m to obtain $J_1\|\ldots\|J_{2\lambda}$
3. Reveal the response for the challenge J_i for each round i of the zero-knowledge proof.
4. Output the signature $\sigma = ((\texttt{com}_i)_i, (\texttt{ch}_{i,j})_{i,j}, (h_{i,j})_{i,j}, (\texttt{resp}_{i,J_i})_i)$

Verification. To verify the signature σ of message m:

1. Compute the hash $J_1\|\ldots\|J_{2\lambda}$
2. Verify the responses for each round of the zero-knowledge proof protocol:
   (a) If $\texttt{ch}_{i,J_i} = 0$, verify that the response is $(R, \phi(R))$ where R and $\phi(R)$ have order $\ell_B^{e_B}$ and generates the kernel of the isogeny ψ and ψ' respectively.
   (b) If $\texttt{ch}_{i,J_i} = 1$, verify that the response has order $\ell_A^{e_A}$ and generates the kernel of the isogeny ϕ'.

Algorithms 3, 4, and 5 gives explicit steps for computing $\texttt{Keygen}, \texttt{Sign}, \texttt{Verify}$.

5.2 Implementation

We now describe some of the lower-level algorithmic and computational aspects of the signature scheme. These techniques were developed in [15] and optimized in [10]. We follow the approach of the latter as our implementation of the signature scheme relies on their SIDH Library.

5.2.1 Choosing Parameters

To find a suitable prime p, we can try various exponents with $\ell_A = 2$ and $\ell_B = 3$, so that roughly $\ell_A^{e_A} \approx \ell_B^{e_B}$ and $\ell_A^{e_A}\ell_B^{e_B} f \pm 1$ is a prime of desired cryptographic size. As in [10], we fix our prime to be

$$p = 2^{372} \cdot 3^{239} - 1$$

Algorithm 3 $\texttt{Keygen}(\lambda)$

```
Pick a random point $S$ of order $\ell_A^{e_A}$
Compute the isogeny $\phi\colon E \to E/\langle S\rangle$
$\texttt{pk} \leftarrow (E/\langle S\rangle, \phi(P_B), \phi(Q_B))$
$\texttt{sk} \leftarrow S$
return $(\texttt{pk}, \texttt{sk})$
```

Algorithm 4 $\texttt{Sign}(\texttt{sk}, m)$

```
for $i = 1$ to $2\lambda$ do
    Pick a random point $R$ of order $\ell_B^{e_B}$
    Compute the isogeny $\psi\colon E \to E/\langle R\rangle$
    Compute either $\phi'\colon E/\langle R\rangle \to E/\langle R, S\rangle$ or $\psi'\colon E/\langle S\rangle \to E/\langle R, S\rangle$
    $(E_1, E_2) \leftarrow (E/\langle R\rangle, E/\langle R, S\rangle$
    $\texttt{com}_i \leftarrow (E_1, E_2)$
    $\texttt{ch}_{i,0} \leftarrow_R \{0, 1\}$
    $(\texttt{resp}_{i,0}, \texttt{resp}_{i,1}) \leftarrow ((R, \phi(R)), \psi(S))$
    if $\texttt{ch}_{i,0} = 1$ then
        $\texttt{swap}(\texttt{resp}_{i,0}, \texttt{resp}_{i,1})$
    $h_{i,j} \leftarrow G(\texttt{resp}_{i,j})$
$J_1\| \dots \|J_{2\lambda} \leftarrow H(\texttt{pk}, m, (\texttt{com}_i)_i, (\texttt{ch}_{i,j})_{i,j}, (h_{i,j})_{i,j})$
return $\sigma \leftarrow ((\texttt{com}_i)_i, (\texttt{ch}_{i,j})_{i,j}, (h_{i,j})_{i,j}, (\texttt{resp}_{i,J_i})_i)$
```

Algorithm 5 $\texttt{Verify}(\texttt{pk}, m, \sigma)$

```
$J_1\| \dots \|J_{2\lambda} \leftarrow H(m, x, (\texttt{com}_i)_i, (\texttt{ch}_{i,j})_{i,j}, (h_{i,j})_{i,j})$
for $i = 1$ to $2\lambda$ do
    check $h_{i,J_i} = G(\texttt{resp}_{i,J_i})$
    if $\texttt{ch}_{i,J_i} = 0$ then
        Parse $(R, \phi(R)) \leftarrow \texttt{resp}_{i,J_i}$
        check $R, \phi(R)$ have order $\ell_B^{e_B}$
        check $R$ generates the kernel of the isogeny $E \to E_1$
        check $\phi(R)$ generates the kernel of the isogeny $E/\langle S\rangle \to E_2$
    else
        Parse $\psi(S) \leftarrow \texttt{resp}_{i,J_i}$
        check $\psi(S)$ has order $\ell_A^{e_A}$
        check $\psi(S)$ generates the kernel of the isogeny $E_1 \to E_2$
if all checks succeed then
    return 1
```

This prime has bitlength 751, providing roughly 124 bits of post-quantum security. The field $\mathbb{F}_{p^2}$ is implemented as $\mathbb{F}_p(i)$ where $i^2 = -1$. The public curve E over $\mathbb{F}_{p^2}$ can be conveniently chosen to be:

$$E : y^2 = x^3 + x$$

which is supersingular and has order $(p+1)^2 = (2^{372} \cdot 3^{239})^2$ [28, Exercise 5.4 & 5.10(a)].

To choose generators (P_B, Q_B) for the torsion subgroup $E[\ell_B^{e_B}]$, we can try picking random points R and checking whether $P_B := [\ell_A^{e_A}]R$ has order $\ell_B^{e_B}$, which will succeed with high probability. We can choose Q_B in the same manner, and check that P_B, Q_B are independent by computing and checking that the Weil pairing $e(P_B, Q_B)$ has order $\ell_B^{e_B}$. This also succeeds with high probability. This gives us a basis (P_B, Q_B) generating the torsion subgroup $E[\ell_B^{e_B}]$.

Alternatively, following [10], we can make use of the *distortion map* which is an endomorphism $\tau : E \to E$ mapping a point $(x, y) \to (-x, iy)$. The distortion map has the property that, if P_B is a full-order $\ell_B^{e_B}$-torsion point, then so is $\tau(P_B)$, and with $Q_B = \tau(P_B)$, (P_B, Q_B) generate a large subgroup of $E[\ell_B^{e_B}]$. They do not generate the full torsion subgroup, since P_B, Q_B are not independent, thus they are technically not a basis of $E[\ell_B^{e_B}]$. However, they offer compactness of parameters and simplify implementation while introducing no known vulnerabilities to the security (even if they did, reverting to the first method would not hurt performance significantly). We can pick P_B randomly as before or, for more compactness, define it deterministically as $P_B := [\ell_A^{e_A}](z, \sqrt{z^3 + z})$ where z is the smallest positive integer such that P_B has order $\ell_B^{e_B}$. In our case, we have

$$P_B = [2^{372}](6, \sqrt{6^3 + 6}) \qquad\qquad Q_B = \tau(P_B)$$

This is the approach we will use for the remainder of this section, and we will refer to (P_B, Q_B) as the basis points, even though they do not generate the full torsion subgroup.

5.2.2 Sampling Torsion Points

The special generators allow us to sample random full-order torsion points easily. It is shown in [10] that, for each $m' \in \{1, 2, \ldots, 3^{e_B - 1} - 1\}$, the point $R := P_B + [m']Q_B$ is a full-order $\ell_B^{e_B}$-torsion point generating distinct subgroups. This samples from roughly a fourth of the $\ell_B^{e_B - 1}(\ell_B + 1)$ distinct cyclic subgroups of $E[\ell_B^{e_B}]$ of order $\ell_B^{e_B}$.

To compute $P_B + [m']Q_B$ once m' is chosen, we can use the three-point ladder algorithm given in [15, Algorithm 1], which can compute the linear combination efficiently in constant time, in contrast to the standard double-and-add algorithm.

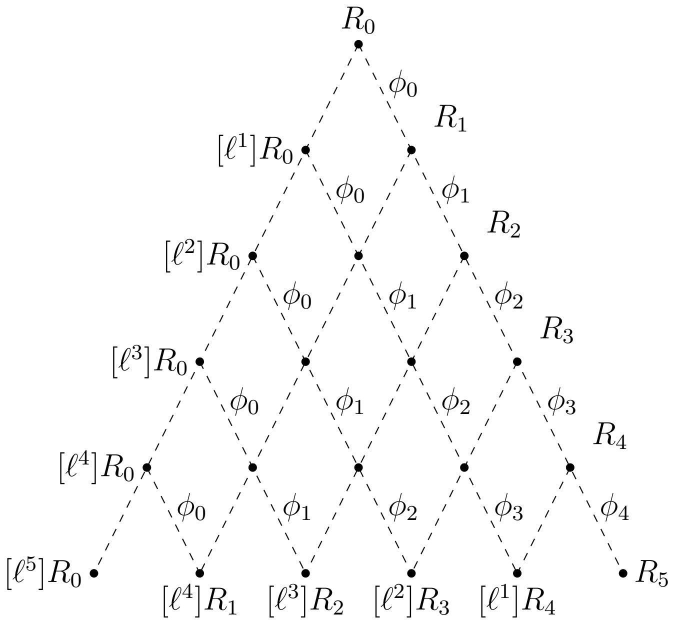

Figure 5.1: A small example illustrating the computation of an isogeny whose kernel is generated by a point R_0 of order ℓ^6.

5.2.3 Computing Isogenies

To compute an isogeny of degree directly using Vèlu's formulas would be impractical, as its runtime is proportional to the size of the kernel which is exponentially large. However, the isogenies in our cryptosystems have degree ℓ^e for some small prime ℓ, and can be computed as a composition e isogenies of degree ℓ, which is more manageable. We will give an overview of this method developed in [15].

Let R be a point of order ℓ^e and suppose we want to compute the isogeny $\phi : E \to E/\langle R \rangle$ whose kernel is generated by R. One possible strategy for decomposing isogeny computations is the following:

1. Set $E_0 = E$ and $R_0 = R$.
2. For $i = 0, \ldots, e-1$:
   (a) Compute the point $[\ell^{e-i-1}]R_i$, which has order ℓ in E_i.

    (b) Use Vèlu's formulas to compute the isogeny $\phi_i : E_i \to E_i/\langle[\ell^{e-i-1}]R_i\rangle$.

    (c) Set $E_{i+1} = E_i/\langle[\ell^{e-i-1}]R_i\rangle$ and $R_{i+1} = \phi_i(R_i)$.

3. Then $E/\langle R\rangle = E_e$ and $\phi = \phi_{e-1} \circ \cdots \circ \phi_0$.

This is called the *multiplication-based* strategy for computing ϕ, which requires a quadratic (in terms of e) number of multiplication-by-ℓ operations, and e evaluations of ℓ-isogenies. Various strategies can be visualized by considering Figure 5.1.

In the figure, the vertices represent points and the dashed edges are directed downwards, representing multiplication-by-ℓ maps and ℓ-isogeny evaluations for leftward and rightward edges, respectively. To compute ϕ, we start with R_0 and compute the necessary edges to compute all points on the bottom line.

There are many different ways (strategies) to do this, as illustrated in Figure 5.2.

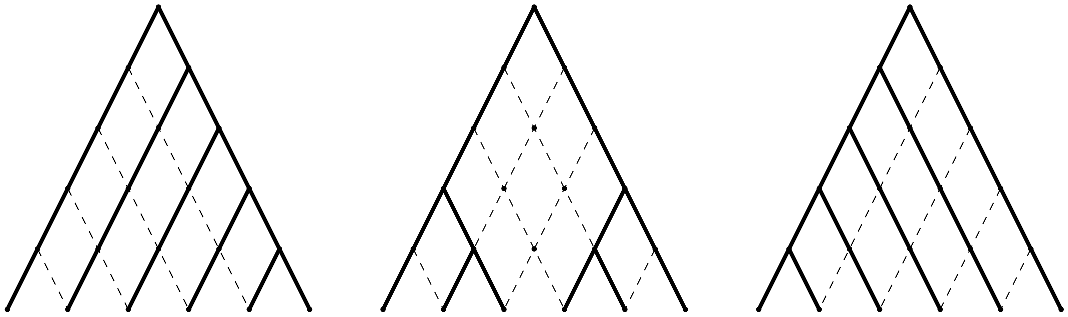

Figure 5.2: Some strategies for computing the isogeny ϕ in Figure 5.1

The left figure represents the multiplication-based strategy described previously. The right figure represents the *isogeny-based strategy* where we compute $[\ell^{e-i-1}]R$ for each $i = 0, 1, \ldots, e-1$ and successively evaluate the isogenies ϕ_j on each point. This strategy requires a quadratic number of isogeny evaluations. While these two strategies require different numbers of multiplication and isogeny evaluations, they have the same total number of operations. However, there are *balanced* strategies which requires fewer operations in total than either of the extreme strategies, as represented by the middle figure.

The most efficient strategy depends on the relative computational costs of the multiplication-by-ℓ map and an ℓ-isogeny evaluation, and can be computed efficiently by dynamic programming. This method is detailed in [15, §4.2.2].

5.2.4 Representing of Curves and Points

We use projective coordinates for both points and curve coefficients as in [10] to reduce the number of field inversions. The curves in our system are isomorphic to Montgomery curves which have the form $E_{(A,B)} : By^2 = x^3 + Ax^2 + x$. The Kummer line on a Montgomery curve, which identifies each point $(X : Y : Z)$ with its inverse $(X : -Y : Z)$, has efficient point arithmetic and allows us to disregard the Y coordinate in our computations. This allows us to represent points by just one field element X/Z in $\mathbb{F}_{p^2}$. However, to compute linear combinations we require an additional x-coordinate of $P - Q$ to perform *differential addition*. We thus include the x-coordinate of $\phi(P_B - Q_B)$ as part of the public key. Isogeny computations are unaffected because a point R and its inverse $-R$ generate the same subgroup.

In the Montgomery form, it turns out that there are only two isomorphism classes of Montgomery curves for a given coefficient value A, and they have the same Kummer line. So the B coefficient also does not affect our computations, and curves can also be represented by one field element for their A-coordinate.

5.3 Parameter Sizes

Recall that our primes have the form $p = \ell_A^{e_A} \ell_B^{e_B} f \pm 1$ with roughly $\ell_A^{e_A} \approx \ell_B^{e_B}$, and that p has bitlength 6λ for λ bits of post-quantum security. So we have $\ell_A^{e_A} \approx \ell_B^{e_B} \approx 2^{3\lambda}$.

Since our curves are defined over $\mathbb{F}_{p^2}$, each field element requires 12λ bits. Curves are represented in Montgomery form $By^2 = x^3 + Ax^2 + x$ where the A-coefficient alone suffices for isogeny computations. Similarly, a point on the Kummer line can be represented by their X-coordinate. In both cases, we need one field element, requiring 12λ bits.

Compression

Recall the compression technique described in §3.4 which compresses curves to their j-invariants and torsion points to their coefficients with respect to a certain basis. Since we are using the Montgomery form of curves which can be represented by one coefficient, the first technique does not reduce our curve sizes. The second technique does reduce the sizes of points, since each coefficient requires 3λ bits and a straightforward representation of a point requires a field element of 12λ bits.

We can apply the compression to our signature scheme in two ways: first to the public key and second to the responses $\psi(S)$ for the rounds where $ch = 1$. The private key and the other responses $(R, \phi(R))$ are already generated using a 3λ-bit coefficient and as such do not require additional computation for compression.

Public Keys

The public key has the form $\mathtt{pk} = (a, x(\phi(P_B)), x(\phi(Q_B)), x(\phi(P_B - Q_B)))$, where a denotes the A-coefficient of the public curve $E/\langle S \rangle$. These four field elements require 48λ bits of storage.

We can compress the public key significantly by compressing the torsion basis $(\phi(P_B), \phi(Q_B))$, requiring three 3λ-bit coefficients. Moreover, the X-coordinate of $\phi(P_B - Q_B)$ is no longer required since the full coordinates of $\phi(P_B)$ and $\phi(Q_B)$ can be recovered from their compressed coefficients. Thus the compressed public key requires 12λ bits for the curve and 9λ bits for the generators, for a total of 21λ bits.

Private Keys

The private key S can be stored as a single coefficient n with respect to a $\ell_A^{e_A}$-torsion basis P_A, Q_A (i.e. $S = P_A + [n]Q_A$), requiring 3λ bits.

Signatures

The signature contains $(\mathtt{com}_i, \mathtt{ch}_{i,j}, h_{i,j}, \mathtt{resp}_{i,J_i})$ for each round i of the ZKP protocol. Each commitment contains two curves (E_1, E_2), each requiring one field element. We need one bit to indicate the first challenge bit $\mathtt{ch}_{i,0}$. We do not need to send $\mathtt{ch}_{i,1}$ since $\mathtt{ch}_{i,1} = 1 - \mathtt{ch}_{i,0}$. The hash $h_{i,j} = G(\mathtt{resp}_{i,j})$ should have bitlength 3λ (this will be justified in §5.4). Note that we do not need to send h_{i,J_i} since it can be computed from $\mathtt{resp}_{i,J_i}$.

The response has a different length depending on the challenge bit J_i. If $J_i = 0$, the response $(R, \phi(R))$ can be represented by their coefficients with respect to the public bases at no additional computational cost, requiring only 3λ bits. If $J_i = 1$, the response $\psi(S)$ requires 12λ bits as a field element. With compression, $\psi(S)$ can be represented in 3λ bits.

In total, each round of the ZKP requires roughly $24\lambda + 1 + 3\lambda + \frac{3\lambda + 12\lambda}{2} \approx 34.5\lambda$ bits on average without compression, and roughly 30λ bits on average with compression. Although λ rounds of the ZKP sufficed for λ bits of post-quantum security, the signature requires 2λ rounds of the ZKP protocol due to the challenge hash being vulnerable to Grover's algorithm [19] (see §5.4.1). So the entire signature has size roughly $69\lambda^2$ ($60\lambda^2$ compressed) bits on average.

For instance, to achieve 128 bits of post-quantum security, our signature scheme requires $48\lambda = 6144$ bits (768 bytes) for the public key (336 bytes compressed), $3\lambda = 384$ bits (48 bytes) for the private key, and $69\lambda^2 = 1,130,496$ bits (141,312 bytes) for the signature (122,880 bytes compressed) on average.

5.3.1 Comparison

We compare our parameter sizes with various post-quantum signature schemes: the stateless hash-based signature SPHINCS-256 [4], a code-based signature based on Niederreiter's variant of the McEliece cryptosystem [5, 11], a lattice-based signature BLISS [14], a recent ring-LWE-based signature TESLA# [3], and the multivariate polynomial-based Rainbow signature [13, 24].

Table 5.1: Comparison of parameter sizes (in bytes) with various post-quantum signature schemes at the quantum 128-bit security level.

Scheme	Public-key size	Private-key size	Signature size
Hash-based	1,056	1,088	41,000
Code-based	192,192	1,400,288	370
Lattice-based	7,168	2,048	5,120
Ring-LWE-based	7,168	4,608	3,488
Multivariate-based	99,100	74,000	424
Isogeny-based	768	48	141,312
Compressed	336	48	122,880

It is clear from Table 5.1 that our isogeny-based signature achieves very small key sizes relative to the other post-quantum signature schemes. We note that the variants of the Merkle signature scheme can achieve smaller (32 byte) key sizes at the same security level, but require state management. We expect future works in isogenies to improve upon signature sizes and performance to produce more practical signatures with still compact keys.

5.4 Security

Theorems 3.3.1 and 4.4.3 imply that our isogeny-based signature scheme obtained in §5.1 is SUF-CMA. However, one important detail in Unruh's proof is that the quantum random oracle G must have the same domain and range for both response types, so that one can substitute G with a random polynomial and invert hashes in the security proof. In §3.4, we described compression techniques giving us a few variants of our signature scheme with a space-time tradeoff (we could compress the public key, the responses, or both), and in §5.3 we also took G to be a random oracle outputting hashes of bitlength $k \approx 3\lambda$. While Unruh's proof applies directly to our compressed signatures, it is invalid in our uncompressed signature scheme where the responses can have bitlength k or $4k$. In this case, we would need to pad the shorter responses to $4k$ bits to apply Unruh's construction. G should then output hashes of bitlength $4k$ so that the domain and range of G are both equal to $\{0,1\}^{4k}$, increasing signature sizes by roughly $18\lambda^2$ bits.

We show that neither compression nor padding is necessary—the uncompressed signature scheme remains secure when G outputs hashes of bitlength $k \approx 3\lambda$. Let $\mathcal{DS}_u$ denote the uncompressed signature scheme and $\mathcal{DS}_c$ denote the scheme where the responses $\psi(S)$ are compressed.

Theorem 5.4.1. *$\mathcal{DS}_c$ is SUF-CMA in the quantum random oracle model.*

Proof. Since all responses are represented by bitstrings of length k, the security of $\mathcal{DS}_c$ follows from Theorem 4.4.3. □

Theorem 5.4.2. *$\mathcal{DS}_u$ is SUF-CMA in the quantum random oracle model.*

Proof. Suppose there exists a quantum polynomial-time adversary $\mathcal{A}$ breaking the SUF-CMA security of $\mathcal{DS}_u$. We show that, given a classical signing oracle to an instance of $\mathcal{DS}_c$ with quantum random oracle $G_c\colon \{0,1\}^k \to \{0,1\}^k$, we can forge a new valid message-signature pair for $\mathcal{DS}_c$ using $\mathcal{A}$.

Suppose we are given the public key `pk` and a signing oracle to an instance of $\mathcal{DS}_c$ with quantum random oracles G_c and H. Let C_0, C_1 denote the set of possible responses to the challenge $ch = 0, 1$ respectively in $\mathcal{DS}_c$. Note that both sets have cardinality roughly 2^k and consist of k-bitstrings. We create an instance of $\mathcal{DS}_u$ with the same setup, except the quantum random oracle G_u is to be defined as follows.

Let U_0, U_1 denote the set of possible responses to the challenge $ch = 0, 1$ respectively in $\mathcal{DS}_u$. Then we have $C_0 = U_0$ and $|C_1| = |U_1|$, but the elements of U_1 are $4k$-bitstrings. Let

$\mathcal{C}\colon U_1 \to C_1$ denote the compression map taking the field representation of a point $\psi(S)$ in U_1 to its compressed coefficient representation in C_1. Then $\mathcal{C}$ is a bijection that can be computed efficiently both ways since the compression map is injective and its inverse just computes the linear combination. Let $G'_u\colon \{0,1\}^{4k} \to \{0,1\}^k$ be a quantum random oracle such that $G'_u(z\|x) = G_c(x)$ for all $x \in \{0,1\}^k$, where z denotes the all-zeros string of length $3k$. Define $G_u\colon \{0,1\}^{4k} \to \{0,1\}^k$ where

$$G_u(x) = \begin{cases} G'_u(z\|\mathcal{C}(x)) & \text{if } x \in U_1 \\ G'_u(\mathcal{C}^{-1}(y)) & \text{if } x = z\|y \text{ where } y \in C_1 \\ G'_u(x) & \text{otherwise} \end{cases}$$

Since G_u just permutes the inputs according to the bijection $\mathcal{C}$ (with MSB zero-padding) before applying the quantum random oracle G'_u, it follows that G_u is indistinguishable from G'_u. Hence $\mathcal{A}$ can break $\mathcal{DS}_u$ when instantiated with G_u.

We give $\mathcal{A}$ the same public key $\texttt{pk}$ with quantum random oracles G_u and H. When $\mathcal{A}$ makes a signing query on a message m, we relay it to the $\mathcal{DS}_c$ signing oracle to get back a signature

$$\sigma = ((\texttt{com}_i)_i, (\texttt{ch}_{i,j})_{i,j}, (h_{i,j})_{i,j}, (\texttt{resp}_{i,J_i})_i)$$

where $J_1\|\ldots\|J_t = H(\texttt{pk}, m, (\texttt{com}_i)_i, (\texttt{ch}_{i,j})_{i,j}, (h_{i,j})_{i,j})$ and $h_{i,j} = G_c(\texttt{resp}_{i,j})$. We simply decompress all responses $\texttt{resp}_{i,J_i}$ in σ where $\texttt{ch}_{i,J_i} = 1$, and give this modified σ to $\mathcal{A}$. Since $G_u(\mathcal{C}^{-1}(y)) = G'_u(z\|y) = G_c(y)$ for all $y \in C_1$, and $G_u(x) = G_c(x)$ for all $x \in C_0$ (with MSB zero-padding of input), it follows that the $h_{i,j}$'s are still valid hashes in $\mathcal{DS}_u$ with G_u. Hence the modified σ is a valid signature for m in $\mathcal{DS}_u$.

Therefore we can answer $\mathcal{A}$'s signing oracle queries so that $\mathcal{A}$ can forge a new valid message-signature pair (m, σ) in $\mathcal{DS}_u$. By similar reasoning, we can then re-compress the new signature without recalculating the hashes to obtain a valid message-signature pair for $\mathcal{DS}_c$, contradicting Theorem 5.4.1. □

5.4.1 Number of Rounds

To achieve λ bits of security, the protocol must be run at least $t = 2\lambda$ times, since a quantum adversary can choose arbitrary bits $J_1\|\ldots\|J_t$, compute simulated proofs using $J_1\|\ldots\|J_t$ as challenge, then perform a pre-image search on H using Grover's algorithm [19] to find a message m that will give the required hash. A faster collision attack does not seem to apply since an adversary must know the challenge bits beforehand in order for their simulated proofs to be verifiable with non-negligible probability. Thus to achieve λ bits of security against quantum attacks, our signature scheme runs the zero-knowledge proof $t = 2\lambda$ times.

We have seen that, in the underlying zero-knowledge proof, revealing responses to both challenges $b = 0, 1$ will allow anyone to compute the secret isogeny. Consequently, it is crucial that our signature scheme does not use the same commitment twice. We show that this happens with negligible probability.

Recall that $p = \ell_A^{e_A} \ell_B^{e_B} f \pm 1 \approx 2^{6\lambda}$ with $\ell_A^{e_A} \approx \ell_B^{e_B} \approx 2^{3\lambda}$. There are roughly $\ell_B^{e_B - 1} - 1 \approx 2^{3\lambda}$ distinct cyclic subgroups of $E[\ell_B^{e_B}]$ from which the commitments are chosen randomly. The zero-knowledge protocol is run 2λ times for each signature, so if we sign 2^s messages, we would select $2^{s+1}\lambda$ cyclic subgroups of $E[\ell_B^{e_B}]$ at random. An upper bound on the probability that we will select the same subgroup at least twice is given by the Birthday bound:

$$\frac{2^{s+1}\lambda(2^{s+1}\lambda - 1)}{2 \cdot 2^{3\lambda}} \leq \frac{2^{2s+2}\lambda^2}{2^{3\lambda+1}} \leq \frac{\lambda^2}{2^{\lambda-1}}$$

for $s \leq \lambda$, which is negligible in λ.

5.5 Performance

Performance tests of the uncompressed signature scheme[1] were run on an Intel Xeon E5-2637 v3 3.5 GHz Haswell processor running CentOS v6.8, compiled with GCC v4.4.7. We also present timing results on the high-performance ARM Cortex-A57 processor in both C and an optimized arithmetic library on ASM [23]. The Juno platform provides a combination of Cortex-A57 and Cortex-A53 cores for ARMv8 big.LITTLE technology. However, our software is only benchmarked on a single high-performance Cortex-A57 core to get the most performance-oriented results. The software is compiled with Linaro GCC v4.9.4 on a single core 1.1GHz ARM Cortex-A57 running OpenEmbedded Linux v4.5.0.

The signing and verifying algorithms are easily parallelizable with linear speedup, since the computations required for each round of the ZKP protocol is independent. We have implemented parallelization for the PC platform. The timing results are summarized in Table 5.2.

As noted before, the computing costs in the signing algorithm are incurred almost entirely in the ZKP rounds which can be precomputed offline. With precomputation, the

[1] Source code is available at https://github.com/yhyoo93/isogenysignature

Table 5.2: Performance results (in 10^6 clock cycles) on Intel Xeon E5-2637 v3 3.5 GHz.

Platform	Threads	Keygen	Signing	Verifying
PC	1	63	28,776	19,679
	2	-	14,474	10,042
	4	-	7,449	5,536
ARM (C)	-	1,656	767,928	493,797
ARM (ASM)	-	123	57,092	36,757

signing algorithm simply needs to evaluate a hash function on the data and output the appropriate responses for the signature.

Chapter 6

Conclusion

We presented and implemented a stateless quantum-resistant digital signature scheme based on supersingular elliptic curve isogenies with very small key sizes, useful for post-quantum applications with strict key size requirements. Combined with previous works, these results show that isogenies can provide the full range of public-key cryptographic primitives including key establishment, encryption, and digital signatures. Though our results are promising, further improvements are still needed to bring isogeny-based signatures truly into the realm of practicality.

References

[1] Ambainis, A., Rosmanis, A., Unruh, D.: Quantum attacks on classical proof systems: The hardness of quantum rewinding. In: 55th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2014, Philadelphia, PA, USA, October 18-21, 2014. pp. 474–483 (2014)

[2] Azarderakhsh, R., Jao, D., Kalach, K., Koziel, B., Leonardi, C.: Key compression for isogeny-based cryptosystems. In: Proceedings of the 3rd ACM International Workshop on ASIA Public-Key Cryptography. pp. 1–10. AsiaPKC '16, ACM, New York, NY, USA (2016)

[3] Barreto, P.S.L.M., Longa, P., Naehrig, M., Ricardini, J.E., Zanon, G.: Sharper ring-lwe signatures. Cryptology ePrint Archive, Report 2016/1026 (2016)

[4] Bernstein, D.J., Hopwood, D., Hülsing, A., Lange, T., Niederhagen, R., Papachristodoulou, L., Schneider, M., Schwabe, P., Wilcox-O'Hearn, Z.: Sphincs: Practical stateless hash-based signatures. In: Oswald, E., Fischlin, M. (eds.) 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 I. pp. 368–397. Springer Berlin Heidelberg, Berlin, Heidelberg (2015)

[5] Bernstein, D.J., Lange, T., Peters, C.: Attacking and defending the mceliece cryptosystem. In: Buchmann, J., Ding, J. (eds.) Post-Quantum Cryptography: Second International Workshop, PQCrypto 2008 Cincinnati, OH, USA, October 17-19, 2008 Proceedings. pp. 31–46. Springer Berlin Heidelberg, Berlin, Heidelberg (2008)

[6] Boneh, D., Zhandry, M.: Secure signatures and chosen ciphertext security in a quantum computing world. In: Canetti, R., Garay, J.A. (eds.) Advances in Cryptology — CRYPTO 2013: 33rd Annual Cryptology Conference, Santa Barbara, CA, USA,

August 18-22, 2013. Proceedings, Part II. pp. 361–379. Springer Berlin Heidelberg, Berlin, Heidelberg (2013)

[7] Childs, A., Jao, D., Soukharev, V.: Constructing elliptic curve isogenies in quantum subexponential time. Journal of Mathematical Cryptology 8(1), 129 (Jan 2014)

[8] Cohen, H., Frey, G., Avanzi, R., Doche, C., Lange, T., Nguyen, K., Vercauteren, F.: Handbook of Elliptic and Hyperelliptic Curve Cryptography, Second Edition. Chapman & Hall/CRC, 2nd edn. (2012)

[9] Costello, C., Jao, D., Longa, P., Naehrig, M., Renes, J., Urbanik, D.: Efficient compression of SIDH public keys. Cryptology ePrint Archive, Report 2016/963 (2016)

[10] Costello, C., Longa, P., Naehrig, M.: Efficient Algorithms for Supersingular Isogeny Diffie-Hellman. In: Advances in Cryptology — CRYPTO 2016: 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part I. pp. 572–601. Springer Berlin Heidelberg, Berlin, Heidelberg (2016)

[11] Courtois, N.T., Finiasz, M., Sendrier, N.: How to achieve a mceliece-based digital signature scheme. In: Boyd, C. (ed.) 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. pp. 157–174. Springer Berlin Heidelberg, Berlin, Heidelberg (2001)

[12] Couveignes, J.M.: Hard homogeneous spaces. Cryptology ePrint Archive, Report 2006/291 (2006), http://eprint.iacr.org/2006/291

[13] Ding, J., Schmidt, D.: Rainbow, a new multivariable polynomial signature scheme. In: Ioannidis, J., Keromytis, A., Yung, M. (eds.) Applied Cryptography and Network Security: Third International Conference, ACNS 2005, New York, NY, USA, June 7-10, 2005. Proceedings. pp. 164–175. Springer Berlin Heidelberg, Berlin, Heidelberg (2005)

[14] Ducas, L., Durmus, A., Lepoint, T., Lyubashevsky, V.: Lattice signatures and bimodal gaussians. In: Canetti, R., Garay, J.A. (eds.) Advances in Cryptology — CRYPTO 2013: 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part I. pp. 40–56. Springer Berlin Heidelberg, Berlin, Heidelberg (2013)

[15] Feo, L.D., Jao, D., Plût, J.: Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. Journal of Mathematical Cryptology 8(3) (Jan 2014)

[16] Fiat, A., Shamir, A.: How to prove yourself: Practical solutions to identification and signature problems. In: Proceedings on Advances in cryptology—CRYPTO '86. pp. 186–194. Springer-Verlag, London, UK, UK (1987)

[17] Galbraith, S.D.: Mathematics of Public Key Cryptography. Cambridge University Press, New York, NY, USA, 1st edn. (2012)

[18] Galbraith, S.D., Petit, C., Silva, J.: Signature schemes based on supersingular isogeny problems. Cryptology ePrint Archive, Report 2016/1154 (2016)

[19] Grover, L.K.: A fast quantum mechanical algorithm for database search. In: Proceedings of the Twenty-eighth Annual ACM Symposium on Theory of Computing. pp. 212–219. STOC '96, ACM, New York, NY, USA (1996)

[20] Jao, D., Feo, L.D.: Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. Post-Quantum Cryptography Lecture Notes in Computer Science p. 1934 (2011)

[21] Jao, D., Soukharev, V.: Isogeny-based quantum-resistant undeniable signatures. Post-Quantum Cryptography Lecture Notes in Computer Science pp. 160–179 (2014)

[22] Koziel, B., Azarderakhsh, R., Kermani, M.M., Jao, D.: Post-quantum cryptography on FPGA based on isogenies on elliptic curves. IEEE Trans. on Circuits and Systems 64-I(1), 86–99 (2017), https://doi.org/10.1109/TCSI.2016.2611561

[23] Koziel, B., Jalali, A., Azarderakhsh, R., Jao, D., Kermani, M.M.: NEON-SIDH: Efficient Implementation of Supersingular Isogeny Diffie-Hellman Key Exchange Protocol on ARM. In: Cryptology and Network Security - 15th International Conference, CANS 2016, Milan, Italy, November 14-16, 2016, Proceedings. pp. 88–103 (2016)

[24] Petzoldt, A., Bulygin, S., Buchmann, J.: Selecting parameters for the rainbow signature scheme. In: Sendrier, N. (ed.) Post-Quantum Cryptography: Third International Workshop, PQCrypto 2010, Darmstadt, Germany, May 25-28, 2010. Proceedings. pp. 218–240. Springer Berlin Heidelberg, Berlin, Heidelberg (2010)

[25] Rostovtsev, A., Stolbunov, A.: Public-key cryptosystem based on isogenies. Cryptology ePrint Archive, Report 2006/145 (2006), http://eprint.iacr.org/2006/145

[26] Seshadri, S.M., Chandrasekaran, V.: Isogeny-based quantum-resistant undeniable blind signature scheme. Cryptology ePrint Archive, Report 2016/148 (2016)

[27] Shor, P.W.: Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM J. Comput. 26(5), 1484–1509 (Oct 1997)

[28] Silverman, J.: The Arithmetic of Elliptic Curves. Graduate Texts in Mathematics, Springer New York (2009)

[29] Stolbunov, A.: Constructing public-key cryptographic schemes based on class group action on a set of isogenous elliptic curves. Advances in Mathematics of Communications 4(2), 215–235 (2010), http://aimsciences.org/journals/displayArticlesnew.jsp?paperID=5170

[30] Sun, X., Tian, H., Wang, Y.: Toward quantum-resistant strong designated verifier signature from isogenies. 2012 Fourth International Conference on Intelligent Networking and Collaborative Systems (2012)

[31] Tani, S.: Claw finding algorithms using quantum walk. Theor. Comput. Sci. 410(50), 5285–5297 (2009)

[32] Tate, J.: Endomorphisms of abelian varieties over finite fields. Inventiones Mathematicae 2(2), 134144 (1966)

[33] Unruh, D.: Quantum proofs of knowledge. In: Advances in Cryptology — EUROCRYPT 2012: 31st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cambridge, UK, April 15-19, 2012. Proceedings. pp. 135–152. Springer Berlin Heidelberg, Berlin, Heidelberg (2012)

[34] Unruh, D.: 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. pp. 755–784. Springer Berlin Heidelberg, Berlin, Heidelberg (2015)

[35] Vèlu, J.: Isogénies entre courbes elliptiques. C. R. Acad. Sci. Paris Sér. A-B (1971)

[36] Watrous, J.: Zero-knowledge against quantum attacks. SIAM Journal on Computing 39(1), 25–58 (2009)

[37] Zhandry, M.: Secure identity-based encryption in the quantum random oracle model. In: Proceedings of the 32Nd Annual Cryptology Conference on Advances in Cryptology — CRYPTO 2012 - Volume 7417. pp. 758–775. Springer-Verlag New York, Inc., New York, NY, USA (2012), http://dx.doi.org/10.1007/978-3-642-32009-5_44

[38] Zhandry, M.: A note on the quantum collision and set equality problems. CoRR abs/1312.1027 (2013), http://arxiv.org/abs/1312.1027

[39] Zhang, S.: Promised and distributed quantum search. In: Wang, L. (ed.) Computing and Combinatorics: 11th Annual International Conference, COCOON 2005 Kunming, China, August 16–19, 2005 Proceedings. pp. 430–439. Springer Berlin Heidelberg, Berlin, Heidelberg (2005)