An Isogeny-Based Adaptor Signature Using SQISign

by

Valerie Gilchrist

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, 2022

Author's Declaration

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

Transactions on blockchains can prove very costly, so as a solution to avoid these large costs, schemes involving payment channel networks have been developed. One approach to implementing these off-chain forms of payment securely involves *adaptor signatures*.

Previous work has established a generic construction of adaptor signatures using signature schemes that satisfy a couple of key properties. Unfortunately, most post-quantum signature schemes do not satisfy these properties, meaning more work needs to be done to develop quantum-safe solutions. We introduce a new post-quantum adaptor signature that uses SQISign as its underlying signature. SQISign has been shown to be significantly faster and to require less storage than any other isogeny-based signature, giving our construction potential for significant improvements in the way of efficiency. We give the details of the new scheme, provide proofs of its security, and estimate memory costs.

Acknowledgements

I would like to thank my supervisor, David Jao, for all of his time and effort over the last two years. His mentorship and expertise have been invaluable throughout my degree. I would also like to thank my readers, Alfred Menezes and Douglas Stebila, for their helpful feedback.

Table of Contents

List of Figures

Chapter 1

Introduction

The development of quantum computers poses a real threat to classical cryptography. Ever since the creation of Shor's algorithm, which allows for a quantum adversary to efficiently factor and compute discrete logarithms (among other things) [21], systems such as RSA risk being broken. As a result, post-quantum cryptography—the search for systems that are still safe against quantum attack—has become an active area of research.

Public key systems rely on underlying hard mathematical problems. To make a system that is quantum-safe, this hard problem must still be hard for a quantum adversary. One such example is the Supersingular Isogeny problem. Before we can state the problem, we first briefly discuss some of the vocabulary. We will not give an exhaustive introduction to the field. The interested reader can refer to [22] for details.

1.1 Mathematical Background

Let k be a field. For simplicity, we assume $\operatorname{char} k \neq 2, 3$.

Definition 1.1.1. An *elliptic curve* E is an algebraic curve of genus 1, defined over k, that can be written in the form:

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

for $a, b \in k$, where $4a^3 + 27b^2 \neq 0$. We denote the set of valid pairs $(x, y) \in k^2$ satisfying E as $E(k)$.

We call this way of writing the curve *Weierstrass* form, but we will often also consider curves in *Montgomery* form which is written as:

$$E : By^2 = x^3 + Ax^2 + x$$

The discriminant, $\Delta = -16(4a^3 + 27b^2)$, and j-invariant, $j = 1728 \cdot 4a^3/(4a^3 + 27b^2)$, are values that can be computed for any elliptic curve, and only depend on the coefficients of the curve's equation. Note that the j-invariant is so-called because it is unique to a curve up to isomorphism.

Definition 1.1.2. An elliptic curve E over the field $\mathbb{F}_q$ of characteristic p is *supersingular* if p divides $\#E(\mathbb{F}_q) - q - 1$. Otherwise we say E is *ordinary*.

For our purposes, we will only be concerned with supersingular elliptic curves.

Definition 1.1.3. An *isogeny*, $\varphi : E \to E'$, is a rational map between two elliptic curves that is also a group homomorphism.

- Isogenies give rise to an injection of function fields: $\varphi^* : \bar{K}(E') \to \bar{K}(E)$. The *degree* of an isogeny, denoted $\deg(\varphi)$, is the degree of the finite extension $\bar{K}(E)/\varphi^*\bar{K}(E')$. Note, degrees are multiplicative, that is $\deg(\varphi_0 \circ \varphi_1) = \deg(\varphi_0) \cdot \deg(\varphi_1)$.
- An isogeny is *separable* if $\deg(\varphi) = |\ker(\varphi)|$. There exists a separable isogeny for every finite subgroup G of E of the form $\phi : E \to E'$ where $\ker(\phi) = G$.
- The *dual* of an isogeny, $\hat{\varphi}$, is such that $\varphi \circ \hat{\varphi} = \hat{\varphi} \circ \varphi = [n]$, where $[n]$ denotes the multiplication by $n = \deg(\varphi)$ map.

We say two elliptic curves are *isogenous* if there exists an isogeny from one of the curves to the other.

Going forward, we are only concerned with separable isogenies, which are determined up to isomorphism by their kernels. By constraining ourselves to only use separable isogenies, we are able to define isogenies only by their kernel generator.

We are now ready to state the general isogeny problem that is at the core of all isogeny-based post-quantum cryptography. There are different variations of this problem used, one of which we will consider later on.

Problem 1.1.1. Given two elliptic curves, E, E', over k that are isogenous over k, compute an isogeny $\varphi : E \to E'$ over k.

It is generally agreed that this is a hard problem for both classical and quantum adversaries [2]. Similarly, the following problem is also believed to be hard:

Problem 1.1.2 (Computational Supersingular Isogeny (CSSI) Problem)**.** Let p be a prime of the form $p = l_A^{e_A} l_B^{e_B} f - 1$. Consider two isogenous curves E and $E/\langle G \rangle$ defined over $\mathbb{F}_{p^2}$ where G is a finite, cyclic subgroup of E. Assuming it exists, find an isogeny $\phi : E \to E/\langle G \rangle$ of degree $l_A^{e_A}$ with (cyclic) kernel $\ker \phi = G$. Equivalently, find a generator of order $l_A^{e_A}$ for G.

Lastly, we state the SIDH relation, which offers up more public information but is still conjectured to be hard for both classical and quantum adversaries within its proposed parameter sets.

Problem 1.1.3 (SIDH Relation)**.** Let p, E, $E/\langle G \rangle$ all be as in Problem 1.1.2. Given a basis (P_B, Q_B) of $E[l_B^{e_B}]$, and (P_A, Q_A) a basis of $E[l_A^{e_A}]$, we have that G can be written as $G = [m_A]P_A + [n_A]Q_A$. Find the isogeny $\varphi : E \to E/\langle G \rangle$ (or equivalently, find the integers m_A and n_A) given $p, E, E/\langle G \rangle, \varphi(P_B), \varphi(Q_B)$.

It was from this problem that the first feasible isogeny-based key exchange scheme came about—*Supersingular Isogeny Diffie-Hellman* (SIDH) [14]. Since then, several other key exchange and signature schemes have been published—the relevant one for our study being *SQISign*.

1.2 Cryptographic Background

Before describing the protocol, we review the relevant background material on signatures, blockchain, adaptor signatures, SQISign, SIDH, and the Quantum Random Oracle Model (QROM).

1.2.1 Digital Signatures

A *digital signature* or *signature* is a cryptographic scheme consisting of three algorithms, $\Sigma = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$, intended to prove the authenticity of a message publicly. The KeyGen phase takes in only a security parameter λ and outputs a secret key, public key pair $(\mathsf{sk}, \mathsf{pk})$. The Sign algorithm takes a secret key and message, and outputs a signature σ. Lastly, Verify is used by the person receiving the signature to check that it was indeed from the claimed signer, using the message and public key as input, and outputting $b \in \{0, 1\}$.

Signatures can often be derived from a sigma protocol, which is a simplified version of the underlying identification scheme. A sigma protocol consists of three rounds and requires input from the verifier, although signature schemes often work around the latter requirement using random oracles. The sigma protocol constitutes the proof of knowledge that the prover will use in the signature scheme. To actually implement a sigma protocol as a signature, there are transforms such as the Fiat-Shamir transform or the Unruh transform that generically convert a sigma protocol into a non-interactive protocol. Since all isogeny-based signature schemes derive from sigma protocols, we will usually discuss signature schemes using the corresponding sigma protocols.

1.2.2 Blockchain

A blockchain is a decentralized secure system that stores transactional data. The purpose is to allow users who don't necessarily trust each other to maintain a common ledger, be it for economic, political, or social reasons. Blockchains were originally created for Bitcoin in order to solve the double spending problem in digital money, but have since evolved to provide solutions in many different areas. The driving ethos is to allow all users to participate without relying on a central institution to oversee them.

The idea behind blockchain was first introduced anonymously in [19]. The idea is to keep a ledger of all transactions, and host it on each participant's—or *node's*—local machine. In order for a new transaction to occur it must be verified by a majority of nodes, that is, the nodes will check that the sender has the amount they claim to. This verification (performed by nodes called *miners*) is done for a fee, thereby incentivizing the nodes to uphold and maintain the integrity of the ledger.

The ledger is made up of a chain of blocks, where each block stores a certain amount of information. Once a block is full, a new block is started, where the timestamp for the new block is dependent on a hash of the previous block's time stamp. This ensures a linear progression of blocks, and helps prevent malicious parties from rewriting a previous block.

Blockchain security relies on the ability to authenticate someone's transactions through digital signatures. Public key cryptography, specifically elliptic curve cryptography, is used to digitally sign users' transactions. Elliptic curve signatures have been in widespread use since the mid 1990s but there is reason to believe that in the next 20 to 30 years such schemes may no longer be secure [18]. Elliptic curve cryptosystems, along with many other classical cryptosystems, rely on the mathematical operation of taking discrete logarithms being difficult for a classical computer. A quantum computer, however, could solve this problem efficiently using Shor's algorithm (or variations thereof), leaving the blockchain

open to quantum attacks. Such attacks include editing blocks (faking past transactions) and forging signatures. In the case of Bitcoin, this means a user could spend more than they have, or steal bitcoin from other users.

1.2.3 Adaptor Signatures

Transactions on blockchains can prove very costly, so as a solution to avoid these large costs, schemes involving payment channel networks have been developed. One approach to implementing these off-chain forms of payment involves *adaptor signatures*.

An adaptor signature (AS) extends the notion of a signature scheme by incorporating an additional hard relation for use in payment channels. The additional hard relation, R, is typically the same one used in the underlying signature scheme, Σ, and consists of witness, statement pairs $(y, Y) \in \mathsf{R}$. We use L_{R} to denote the set of valid statements in R.

In an adaptor signature, for any statement Y, a signer holding a secret key can produce a *pre-signature* on any message m. This *pre-signature* can be adapted into a full signature on m if and only if the user has the witness to Y. Additionally, anyone with access to the pre-signature, full signature, and Y, is able to from this extract the witness. We formalize these requirements below, as given in [23].

Definition 1.2.1 (Adaptor Signature Scheme)**.** An adaptor signature scheme with respect to a hard relation R and a signature scheme $\Sigma = (\mathsf{KeyGen}, \mathsf{Sig}, \mathsf{Ver})$ consists of four algorithms:

$\mathsf{PreSig}(\mathsf{sk}, m, Y)$: is a PPT algorithm, where on input of a secret key sk, message $m \in \{0,1\}^*$, and statement $Y \in L_{\mathsf{R}}$, outputs a pre-signature $\widetilde{\sigma}$.
$\mathsf{PreVer}(\mathsf{pk}, m, Y, \widetilde{\sigma})$: is a DPT algorithm that on input of a public key pk, a message $m \in \{0,1\}^*$, a statement $Y \in L_{\mathsf{R}}$, and a pre-signature $\widetilde{\sigma}$, outputs a bit b.
$\mathsf{Adapt}(\widetilde{\sigma}, y)$: is a DPT algorithm that on input of a valid pre-signature $\widetilde{\sigma}$, and a witness y, outputs a signature σ.
$\mathsf{Extract}(\sigma, \widetilde{\sigma}, Y)$: a DPT algorithm that on input of a pre-signature $\widetilde{\sigma}$, a corresponding signature σ, and a statement Y, outputs the witness y to the statement Y.

The adaptor signature will also inherit KeyGen and Ver from Σ. We use GenR to denote the algorithm that generates the secret, statement pair (y, Y).

Some desirable attributes we want in adaptor signatures include having honestly generated pre-signatures always be able to be adapted into correct full signatures (pre-signature

correctness), signatures should be infeasible for an adversary to forge ($\mathsf{aEUF-CMA}$ security), that any verifiable pre-signature (not necessarily honestly generated) can be adapted into correct full signatures (pre-signature adaptability), and that any valid (pre-signature, signature) pair will allow for the extraction of the relevant witness (witness extractability). We now formalize these ideas in security definitions that are required for any valid and secure adaptor signature.

Definition 1.2.2 (Pre-signature correctness)**.** An adaptor signature $\mathsf{aSIG}_{\mathsf{R},\Sigma}$ satisfies *pre-signature correctness*, if for all $n \in \mathbb{N}$ and $m \in \{0,1\}^*$:

$$Pr\left[\begin{array}{l|l} \mathsf{pVrfy}_{pk}(m, Y; \widetilde{\sigma}) = 1 \wedge & (sk, pk) \leftarrow \mathsf{Gen}(1^n), (y, Y) \leftarrow \mathsf{GenR}(1^n) \\ \mathsf{Vrfy}_{pk}(m; \sigma) = 1 \wedge & \widetilde{\sigma} \leftarrow \mathsf{pSign}_{sk}(m, Y), \sigma := \mathsf{Adapt}(\widetilde{\sigma}, y) \\ (Y, y') \in \mathsf{R} & y' := \mathsf{Ext}(\sigma, \widetilde{\sigma}, Y) \end{array}\right] = 1.$$

Definition 1.2.3 ($\mathsf{aEUF-CMA}$ Security)**.** An adaptor signature scheme $\mathsf{aSIG}_{\mathsf{R},\mathsf{SIG}}$ is *unforgeable* if for every PPT adversary $\mathcal{A}$ there exists a negligible function ν such that: $\Pr[\mathsf{aSigForge}_{\mathcal{A},\mathsf{aSIG}_{\mathsf{R},\mathsf{SIG}}}(n) = 1] \leq \nu(n)$, where the definition of the experiment $\mathsf{aSigForge}_{\mathcal{A},\mathsf{aSIG}_{\mathsf{R},\mathsf{SIG}}}$ *is as follows:*

$\mathsf{aSigForge}_{\mathcal{A},\mathsf{aSIG}_{\mathsf{R},\mathsf{SIG}}}(n)$	$\mathcal{O}_S(m)$	$\mathcal{O}_{pS}(m, Y)$
$1: \mathcal{Q} := \emptyset, (sk, pk) \leftarrow \mathsf{KeyGen}(1^n)$	$1: \sigma \leftarrow \mathsf{Sign}_{sk}(m)$	$1: \widetilde{\sigma} \leftarrow \mathsf{pSign}_{sk}(m, Y)$
$2: m^* \leftarrow \mathcal{A}^{\mathcal{O}_S, \mathcal{O}_{pS}}(pk)$	$2: \mathcal{Q} := \mathcal{Q} \cup \{m\}$	$2: \mathcal{Q} := \mathcal{Q} \cup \{m\}$
$3: (y, Y) \leftarrow \mathsf{GenR}(1^n), \widetilde{\sigma} \leftarrow \mathsf{pSign}_{sk}(m^*, Y)$	$3:$ **return** σ	$3:$ **return** $\widetilde{\sigma}$
$4: \sigma^* \leftarrow \mathcal{A}^{\mathcal{O}_S, \mathcal{O}_{pS}}(\widetilde{\sigma}, Y)$		
$5:$ **return** $\left(m^* \notin \mathcal{Q} \wedge \mathsf{Vrfy}_{pk}(m^*; \sigma^*)\right)$		

Definition 1.2.4 (Pre-signature adaptability)**.** An adaptor signature scheme $\mathsf{aSIG}_{\mathsf{SIG},\mathsf{R}}$ satisfies *pre-signature adaptability*, if for all $n \in \mathbb{N}$, messages $m \in \{0,1\}^*$, statement/witness pairs $(y, Y) \in \mathsf{R}$, public keys pk and pre-signatuers $\widetilde{\sigma} \leftarrow \{0,1\}^*$ we have if $\mathsf{pVrfy}_{pk}(m, Y; \widetilde{\sigma}) = 1$, then $\mathsf{Vrfy}_{pk}(m; \mathsf{Adapt}_{pk}(\widetilde{\sigma}, y)) = 1$.

Definition 1.2.5 (Witness extractability)**.** An adaptor signature scheme $\mathsf{aSIG}_{\mathsf{R}}$ is *witness extractable* if for every PPT adversary $\mathcal{A}$, there exists a negligible function ν such that the following holds: $\Pr[\mathsf{aWitExt}_{\mathcal{A},\mathsf{aSIG}_{\mathsf{R},\mathsf{SIG}}}(n) = 1] \leq \nu(n)$, where the experiment $\mathsf{aWitExt}_{\mathcal{A},\mathsf{aSIG}_{\mathsf{R},\mathsf{SIG}}}$ is defined as follows:

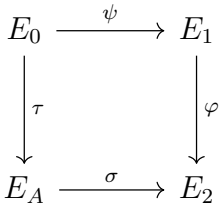

Figure 1.1: SQISign identification protocol.

$\mathsf{aWitExt}_{\mathcal{A},\mathsf{aSIG}_{\mathsf{R},\mathsf{SIG}}}(n)$	$\mathcal{O}_S(m)$	$\mathcal{O}_{pS}(m, Y)$
$1 : \mathcal{Q} := \emptyset, (sk, pk) \leftarrow \mathsf{Gen}(1^n)$	$1 : \sigma \leftarrow \mathsf{Sign}_{sk}(m)$	$1 : \widetilde{\sigma} \leftarrow \mathsf{pSign}_{sk}(m, Y)$
$2 : (m^*, Y^*) \leftarrow \mathcal{A}^{\mathcal{O}_S, \mathcal{O}_{pS}}(pk)$	$2 : \mathcal{Q} := \mathcal{Q} \cup \{m\}$	$2 : \mathcal{Q} := \mathcal{Q} \cup \{m\}$
$3 : \widetilde{\sigma} \leftarrow \mathsf{pSign}_{sk}(m^*, Y^*)$	$3 :$ **return** σ	$3 :$ **return** $\widetilde{\sigma}$
$4 : \sigma^* \leftarrow \mathcal{A}^{\mathcal{O}_S, \mathcal{O}_{pS}}(\widetilde{\sigma})$		
$5 : y := \mathsf{Ext}_{pk}(\sigma^*, \widetilde{\sigma}, Y^*)$		
$6 :$ **return** $\left(m^* \notin \mathcal{Q} \wedge (Y^*, y) \notin \mathsf{R} \wedge \mathsf{Vrfy}_{pk}(m^*; \sigma^*)\right)$		

1.2.4 SQISign

SQISign [8] is an isogeny-based signature scheme that was published in 2020. It is, to date, the most compact post-quantum signature, and its speed and functionality have recently been further improved by De Feo et al. [13]. It uses a single iteration of its identification protocol, which is outlined in Figure 1.1.

The prover starts with a secret isogeny, τ. They publish its codomain, E_A, as their public key and keep τ as their secret key. They randomly select another map, ψ, with domain E_0 as their commitment, and only send its codomain, E_1, to the verifier. The commitment ψ must remain just as secret as τ. The verifier sends back a map, φ, with domain E_1 as a challenge. In practice, the challenge is created using a hash of the message. To prove knowledge of the secret, the prover sends an isogeny from E_A to the challenge codomain, E_2. They do so by composing the dual of the secret, $\hat{\tau}$, along with ψ and φ, and running an algorithm called KLPT on the resulting map. This gives a single isogeny map, σ, of fixed degree, and appropriate domain, and codomain, without leaking any information about τ or ψ.

Note: The prover must know all of the isogenies in the chain in order to run KLPT and get back a single isogeny.

The **KLPT** algorithm, named after its inventors, is at the heart of SQISign, and was first introduced in [16], but later improved upon in the actual SQISign paper [8]. It uses the Deuring Correspondence to create a mapping between the relevant isogenies and left ideals in the quaternion algebra $\mathcal{B}_{p,\infty} = \mathbb{Q}[i,j]$. Via this correspondence, finding a single isogeny equivalent to the composition of three isogenies becomes the problem of how to find an ideal equivalent to the multiplication of three other ideals. Below, we summarize the central step of the algorithm, which is: given an ideal I of norm $n(I)$, find an equivalent ideal J of a fixed norm N. First, we require the following lemma:

Lemma 1.2.1. *Let I be an integral ideal. Then we have the following surjection from $I \setminus \{0\}$ to the set of ideals equivalent to I:*

$$\chi_I(\alpha) = I\frac{\bar{\alpha}}{n(I)}$$

where $\chi_I(\alpha) = \chi_I(\beta)$ only when $\alpha = \beta\delta, \delta \in \mathcal{O}_R(I)^\times$, $\mathcal{O}_R$ a maximal order.

The proof of this lemma can be found in both [8] and [16].

KLPT first takes a left $\mathcal{O}_0$-ideal I as input, where $\mathcal{O}_0$ is a maximal order of $\mathcal{B}_{p,\infty}$. It then takes $\delta \in I$ and uses the value $\chi_I(\delta)$ from Lemma 1.2.1 to obtain an equivalent ideal, L. Next, it searches L to find an element β of the desired degree. The algorithm then returns $J = \chi_L(\beta)$.

In the SQISign paper [8] the KLPT algorithm is further generalized to arbitrary maximal ideals (not just $\mathcal{O}_0$, which corresponds to the base curve E_0 used in protocols) which was the key contribution of the paper that allowed for the signature scheme to operate. Furthermore, the norm of the outputs was decreased as well. These improvements are achieved by taking advantage of *Eichler orders* and *special extremal orders* to narrow the set of elements over which they must search to find the desired β element previously mentioned, making the algorithm feasible in the generalized setting.

1.2.5 Supersingular Isogeny Diffie-Hellman (SIDH)

Supersingular Isogeny Diffie-Hellman (SIDH) is a Diffie-Hellman-like key exchange system first proposed by Jao, De Feo, and Plût [14] meeting post-quantum security standards. It was later submitted to the United States National Institute of Standards and Technology's

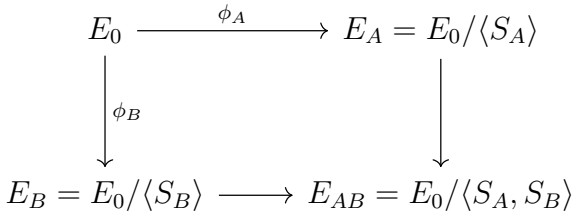

Figure 1.2: An SIDH square.

(NIST) standardization process for post-quantum cryptography systems, under the name of Supersingular Isogeny Key Encapsulation (SIKE), and is currently under consideration in the third round of the process as an alternate candidate. We give a brief summary of the system. For further details the reader can reference [14].

We begin with two parties, Alice and Bob, who would like to arrive at a shared secret. They first agree on a prime p of the form $\ell_A^{e_A}\ell_B^{e_B}f \pm 1$, where ℓ_A, ℓ_B are small primes. In practice, ℓ_A and ℓ_B are generally taken to be 2 and 3 respectively. The size of p will depend on a security parameter λ. Alice and Bob fix the field of definition to be $\mathbb{F}_{p^2}$, and select a base curve E_0. They each select a secret isogeny, ϕ_A, ϕ_B, respectively from the base curve, and publish the codomain of their secrets for use as their public keys, E_A, E_B, respectively. We are assured that Alice's secret isogeny is separable and of degree $\ell_A^{e_A}$, similarly Bob's secret will have degree $\ell_B^{e_B}$.

Since the isogenies are separable we can represent them by their kernels, and hence their kernel generators. We let the kernel of ϕ_A be $S_A = \langle P_A + \alpha Q_A \rangle$, and let $\ker(\phi_B) = S_B = \langle P_B + \beta Q_B \rangle$, where $\alpha \in \mathbb{Z}/\ell_A^{e_A}\mathbb{Z}, \beta \in \mathbb{Z}/\ell_B^{e_B}\mathbb{Z}$ and (P_A, Q_A), (P_B, Q_B) are generating sets for $E[\ell_A^{e_A}]$ and $E[\ell_B^{e_B}]$ respectively. Alice and Bob will each publish these generating sets along with their public keys, so their secrets simplify to the integers α, β.

Once the keys have been generated and the public information shared, Alice will compute $\phi_A(P_B)$ and $\phi_A(Q_B)$ and send it to Bob. Bob will now compute the shared curve E_{AB} with his secret integer β as $E_{AB} = E_A/\langle \phi_A(P_B) + \beta\phi_A(Q_B) \rangle = E_A/\langle \phi_A(P_B + \beta Q_B) \rangle$. Alice will take a similar approach to arrive at $E_{BA} = E_B/\langle \phi_B(P_A) + \alpha\phi_B(Q_A) \rangle = E_B/\langle \phi_B(P_A + \alpha Q_A) \rangle = E_{AB}$. The commutative square diagram is depicted in Figure 1.2.

1.2.6 Quantum Random Oracle Model (QROM)

Since we are working towards a quantum-safe adaptor signature, we will assess security in the Quantum Random Oracle Model (QROM) as introduced in [24]. The model uses

the Unruh transform to adapt any interactive sigma protocol into a non-interactive zero knowledge (NIZK) proof, with the additional *online extractability* property. This property ensures that the witness from a proof can be extracted assuming access to the random oracle.

In the quantum setting, we have that the adversary submitting queries to the oracle has the choice to submit queries in superposition, meaning that reading the queries they submit would change the state of the original query. In order to address this problem of reading queries, an efficient method of inverting the query calls is used to determine the inputs of two calls, which is enough to extract the witness being used. See [24] for full details on this technique.

At a high level, the prover will send their responses to the possible challenges $b \in \{0, 1\}$ to the oracle, obtaining a hash of each response. The verifier can then send in their challenge, and receive back a hash of the prover's appropriate response. Note that the oracle will require the prover to submit several proofs, and so at least a large portion of them will need to be valid. Thus, any proofs selected to be inverted will be valid with a high probability, enabling the extraction of the witness.

The zero-knowledge proof consists of two polynomial-time algorithms (P, V). We assume the statement, witness pair Y, y respectively, are valid inputs of the hard relation R. By definition, a random oracle, H, involves a probability distribution on a set of functions; denote this distribution by ROdist. Lastly, we let π_y be a proof of Y produced by P. We require that $P^H(y, Y) = \perp$ whenever $(y, Y) \notin \mathsf{R}$ and that $V^H(Y, \pi_y) \in \{0, 1\}$.

We list the security definitions that our zero-knowledge proof in SQI-AS will satisfy below.

Definition 1.2.6 (Completeness)**.** $P(Y, y), V(Y, \pi_y)$ is complete if and only if for any quantum-polynomial-time oracle algorithm A, we have that

$$Pr[(Y, y) \in \mathsf{R} \wedge ok = 0 : H \leftarrow \mathsf{ROdist}, (Y, y) \leftarrow A^H(), \pi \leftarrow P^H(Y, y), ok \leftarrow V^H(Y, \pi)]$$

is negligible.

Definition 1.2.7 (Zero-knowledge)**.** Given a simulator $(\mathcal{S}, \mathcal{H})$, the oracle $\mathcal{H}'(Y, y)$ does: if $(Y, y) \notin \mathsf{R}$, return $\perp$; else return $\mathcal{H}(Y)$. (The purpose of $\mathcal{H}'$ is merely to serve as an interface for the adversary who expects a prover taking two arguments Y, y.)

A non-interactive proof system (P,V) is zero-knowledge if and only if there is a polynomial-time simulator $(\mathcal{S}, \mathcal{H})$ such that for every quantum-polynomial-time oracle algorithm A, we have that

$$|Pr[b = 1 : H \leftarrow \mathsf{ROdist}, b \leftarrow A^{H,P}()] - Pr[b = 1 : H \leftarrow \mathcal{S}(), b \leftarrow A^{H,\mathcal{H}'}()]|$$

is negligible.

We assume that both $\mathcal{S}$ and $\mathcal{H}$ have access to and may depend on a polynomial upper bound on the runtime of A.

Definition 1.2.8 (Online extractability)**.** A non-interactive proof system (P,V) is online extractable with respect to $\mathcal{S}$ if and only if there is a polynomial-time extractor E such that for any quantum-polynomial-time oracle algorithm A, we have that

$$Pr[ok = 1 \wedge (Y, y) \notin \mathsf{R} : H \leftarrow \mathcal{S}(), (Y, \pi_y) \leftarrow A^H(), ok \leftarrow V^H(Y, \pi_y), y \leftarrow E(H, Y, \pi_y)]$$

is negligible.

We assume that both $\mathcal{S}$ and E have access to and may depend on a polynomial upper bound on the runtime of A.

The transformation of any sigma protocol with the honest verifier zero-knowledge and special soundness properties into a NIZK proof in QROM with the preceding properties is described in detail in [24].

1.3 State of the Art

Adaptor signatures were first introduced in [3], giving examples using only signatures of classical security. More recently, a paper on how to generically create adaptor signature schemes from classical signatures meeting two particular criteria was published [10]. One of these criteria require that the signature scheme satisfy a homomorphic property that not all post-quantum signatures meet. In particular, SQISign does not meet it.

There are currently only two other post-quantum adaptor signatures that have been published. The first being IAS [23], an isogeny-based protocol using CSI-FiSh as its underlying signature scheme. The second is LAS [11], a lattice-based system. We now discuss both in detail.

1.3.1 IAS

Isogeny Adaptor Signature (IAS) [23] is the only other isogeny-based adaptor signature to be published, using CSI-FiSh [4] as its underlying signature scheme. The paper introduces the protocol itself, proofs of its security, how it might be applied in a blockchain setting, and a memory and efficiency analysis. We summarize the structure of the scheme here.

CSI-FiSh. Commutative Supersingular Isogeny based Fiat-Shamir signatures (CSI-FiSh) [4] is an isogeny-based signature scheme that was developed in the wake of Commutative Supersingular Isogeny Diffie-Hellman (CSIDH) [5]. It uses the same commutative class group action as CSIDH given by

$$\star : Cl(\mathcal{O}) \times \mathcal{A} \to \mathcal{A}$$

$$\mathbf{a} \star E_0 \mapsto E_a$$

where $\mathcal{A}$ is the set of isomorphism classes of feasible elliptic curves, and $Cl(\mathcal{O})$ is the set of ideal classes over a curve's endomorphism ring. In particular, the group action will use the set of ideal classes of E_0, which form a group. Since each ideal can be thought of as a subset of endomorphisms, which can be defined by a single kernel set, we represent the ideal *class* by the intersection of these kernels. This in turn leaves us with a new kernel set, which gives rise to a homomorphism. Note this is no longer an endomorphism, so it maps our starting curve into a new elliptic curve, E_a.

We now describe a single iteration of the sigma protocol for CSI-FiSh. Both parties begin by agreeing on a starting curve E_0. The prover then begins by choosing their secret key $\mathbf{a} \in Cl(\mathcal{O})$ and publishing $\mathbf{a} \star E_0 = E_a$ as their public key. Similarly they choose their commitment uniformly at random $\mathbf{b} \in Cl(\mathcal{O})$. Note that one of the contributions of [4] was establishing how to make these selections uniformly at random by selecting a standard generator of $Cl(\mathcal{O})$, and choosing integers at random as exponents for the generator. Therefore, for $b \in \mathbb{Z}$ and generator $\mathbf{g}$ of $Cl(\mathcal{O})$, the class group action can be written as

$$\mathbf{g}^b \star E = [b]E$$

The prover will publish $[b]E_0 = E_b$ as their commitment.

The verifier will then issue a challenge $c \in \{0, 1\}$. If $c = 0$, the prover will reveal the commitment $\mathbf{b}$ and the verifier will check that this gives a map from E_0 to E_b. If $c = 1$, the prover will reveal $\mathbf{ba^{-1}}$, and the verifier will check that this indeed gives a map from E_a to E_b. With enough iterations the verifier will be assured of the prover's knowledge. This protocol is depicted in Figure 1.3.

IAS. IAS uses an isogeny-based witness/statement set, where the witnesses are elements $\mathbf{y} \in Cl(\mathcal{O})$ and the statements are curves $\mathbf{y} \star E_0 = E_Y$.

After fixing a witness/statement pair, the pre-signer will choose a random $b \in \mathbb{Z}$, and compute $\hat{E}' = [b]E_0$ and $E' = [b]E_Y$. E' is then used to canonically determine the challenge, c, and is included as part of the final pre-signature so that later parties can verify

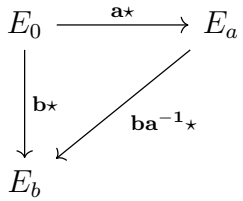

Figure 1.3: Sigma protocol of CSI-FiSh.

the challenge was computed honestly. Additionally, a zero-knowledge proof π showing that the same b was used in $\hat{E}'$ and E' is provided. Therefore the final pre-signature to be sent is $\hat{\sigma} = (\hat{r}, c, \pi, E')$, where $\hat{r}$ is a response that is computed depending on c as described in the CSI-FiSh protocol. To adapt $\hat{\sigma}$ into a full signature, the user computes $r = y + \hat{r}$, and uses $\sigma = (r, c)$ as the signature. Extraction is achieved simply by subtracting $y = r - \hat{r}$.

Verification of the pre-signature is done by recomputing $\hat{E}' = [\hat{r}]E_c$ where E_c is either E_a or E_0 depending on the challenge, and then checking that π is correct and that the challenge was generated honestly. Verification of the full signature is inherited from CSI-FiSh.

Due to the reliance on CSI-FiSh, IAS has limitations in its parameter sizes. CSI-FiSh can run on at most the CSIDH-512 parameters, due to the fact that knowledge about the structure of the class group is needed in order to compute the class group action on uniformly random group elements. The best known algorithm to do so is classically sub-exponential, making larger parameter sets currently not viable to compute.

SeaSign [7] is an isogeny-based signature that also uses the class group action of CSIDH, but that replaces the uniformly random group element sampling with rejection sampling. This change allows the signature to circumvent the requirement to complete the pre-computation of the class group action structure in CSI-FiSh. If IAS were to use a signature such as SeaSign, it might be able to improve the parameter sizes but this would then result in an even slower run time and a more complicated security analysis.

1.3.2 LAS

LAS [11], a lattice-based adaptor signature, was the first published post-quantum adaptor signature. It relies on standard lattice assumptions such as Module-LWE and Module-SIS, and uses a simplified version of Dilithium as its underlying signature [9]. The paper

summarizes the protocol and discusses its security, gives details on how to implement the system on a blockchain, and gives efficiency analyses.

For the sake of the adaptor signature scheme, it is sufficient to simply understand that it is generally difficult to find a short vector v, given a particular matrix A, such that $Av = 0$. For the curious reader, we discuss the technical details in the **Signature** section first.

Signature. As mentioned, the signature scheme to be used is a simplified version of Dilithium, where some of the optimizations are not used for ease of presentation. The protocol operates on the conjecturally hard Problem 1.3.1 and Problem 1.3.2, stated below.

Define $\mathcal{R}_q = \mathbb{Z}_q[X]/(X^d + 1)$ to be a cyclotomic ring with degree $d = 2^l$ for $l \in \mathbb{Z}$, and an odd modulus q. Let $\mathbb{S}_c$ denote the set of polynomials in $\mathcal{R}_q$ whose maximum absolute coefficient is $c \in \mathbb{Z}^+$.

Define $\mathbf{I}_n$ as the nth-dimensional identity matrix.

Problem 1.3.1 (M-SIS)**.** Choose $m > n > 0$. Let $\mathbf{A}' \in \mathcal{R}_q^{n\times(m-n)}$. Let $\mathbf{A} = [\mathbf{I}_n||\mathbf{A}']$. Find a short, non-zero $\mathbf{v} \in \mathcal{R}_q^m$ such that $\mathbf{Av} = 0$ over $\mathcal{R}_q$ and $||\mathbf{v}||$ be less than a fixed amount $\beta_{SIS} < q$.

Problem 1.3.2. Fix $m, l > 0$. Distinguish between the following two cases:

1. $(\mathbf{A}, \mathbf{b}) \in \mathcal{R}_q^{m\times l} \times \mathcal{R}_q^m$
2. $(\mathbf{A}, \mathbf{As} + \mathbf{e})$ for $\mathbf{A} \in \mathcal{R}_q^{m\times l}$, a secret vector $\mathbf{s} \in \mathbb{S}_1^l$, and an error vector $e \in \mathbb{S}_1^m$.

We can now consider the Dilithium protocol by outlining one iteration. Note that to prevent leaking information, both LAS and Dilithium use the rejection sampling technique. That is, in order for the sampling of the masked vector, which will be our response, $\mathbf{z} = \mathbf{y}+\mathbf{s}$ to be simulatable as random we sample $\mathbf{y}$ from $\mathbb{S}_\gamma^k$ for $\gamma \approx kd \cdot p$, and reject $\mathbf{z}$ if its norm is not of sufficiently small size. This results in a uniform distribution of $\mathbf{z}$ over $\mathbb{S}_{\gamma-p}^k$, as desired.

To begin, the prover generates $\mathbf{A} \in \mathcal{R}_q^{k\times l}$, as well as secret key short vectors $\mathbf{s_1}$ and $\mathbf{s_2}$. They publish $\mathbf{A}$ and $\mathbf{t} = \mathbf{As_1} + \mathbf{s_2}$.

The prover selects a masking vector $\mathbf{y}$, and computes $\mathbf{w_1}$ to be the "high order" bits of the coefficients of $\mathbf{Ay}$. The challenge c is computed as a polynomial in $\mathcal{R}_q$, which will depend on the message and $\mathbf{w_1}$ via a hash function. The response will be $\mathbf{z} = \mathbf{y} + c\mathbf{s_1}$. If $\mathbf{z}$ is not accepted during the rejection sampling step, then another $\mathbf{y}$ is chosen, repeating this until $\mathbf{z}$ is accepted. The prover sends $\mathbf{z}$ to the verifier.

To verify, the verifier will first compute $\mathbf{w_1'}$, the "high order" bits of $\mathbf{Az} - c\mathbf{t}$. Then they will check that the challenge generated by the message and $\mathbf{w_1'}$ matches the challenge they were sent. Lastly they will check that the norm of $\mathbf{z}$ is as expected.

LAS. The witness/statement pair set we are drawing from are of the form (y, Y) where $Y = \mathbf{A}y$ for a fixed $\mathbf{A} \in \mathcal{R}_q^{n\times(n+l)}$ and for a short y. Similarly, the general (public key, secret key) pair used in signing is $(\mathbf{t}, \mathbf{r})$ where $\mathbf{t} = \mathbf{Ar}$.

To presign, the prover first selects a masking vector $\mathbf{y} \in \mathbb{S}_\gamma^{n+l}$, and then computes $\mathbf{w} = \mathbf{Ay}$. From this, they define the challenge to be a hash of $\mathbf{t}, \mathbf{w} + Y$, and the message. Recall that $(\mathbf{t}, \mathbf{r})$ is the public key, secret key pair. Next, they compute $\hat{\mathbf{z}} = \mathbf{y} + c\mathbf{r}$, and repeat this process until $\hat{\mathbf{z}}$ is accepted in the rejection sampling step. They publish the presignature $\hat{\sigma} = (c, \hat{\mathbf{z}})$.

To adapt the presignature into a full signature, we simply return $\sigma = (c, \hat{\mathbf{z}} + y)$, where y is the witness. Hence, if we are provided with σ and $\hat{\sigma}$, we can extract the witness $y = \mathbf{s} - \hat{\mathbf{z}}$.

In practice, this adaptor signature is expected to run almost as quickly as a regular lattice-based signature, with some loss due to the absence of some of the optimizations. Security levels and parameter sizes are expected to be the same as those suggested by Dilithium.

Chapter 2

A New Isogeny-Based Adaptor Signature

2.1 SQI-AS

In the previous work included in [10], a generic construction is described to convert any signature scheme satisfying a particular homomorphic property and that is commitment-recoverable into an adaptor signature. Unfortunately, most post-quantum signature schemes seem to not satisfy these properties, meaning more work needs to be done to develop quantum-safe solutions. In the works of [23] and [11], post-quantum adaptor signatures are introduced using isogeny and lattice based underlying signatures respectively. We now introduce a new post-quantum adaptor signature that uses SQISign as its underlying signature, which we will refer to henceforth as *SQI-AS*. SQISign has been shown to be significantly faster and requiring less storage than any other isogeny-based signature, along with more flexibility in its parameter sets, giving our construction potential for significant improvements in the way of efficiency and security.

To use SQISign as the underlying signature scheme in an adaptor signature, we select our hard relation, R_{SSI}, based on the Computational Supersingular Isogeny (CSSI) hard problem 1.1.2 and the SIDH Relation 1.1.3. We will also choose our base curve E_0 to match the requirements laid out in SQISign [8]. Thus, we define our hard relation to be the set

$$\mathsf{R}_{SSI} := \{\big(y, (E_Y, \pi_Y)\big) | y : E_0 \to E_Y, E_Y = E_0/\langle P_0 + \alpha Q_0\rangle, \alpha \in \mathbb{Z}\},$$

where π_Y is a zero-knowledge proof that (y, E_Y) are a valid pair of the hard relation R_{SSI}.

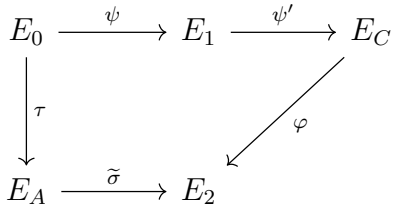

Figure 2.1: PreSign algorithm.

In other words, y (the witness) is an isogeny from the base curve E_0 to the curve E_Y. The elliptic curve parameters of E_0 will include a pair (P_0, Q_0) which will serve as a basis to compute the kernel of y, hence the secret of y can be reduced down to a single integer, i.e. the linear combination of P_0 and Q_0 used to compute $\ker(y)$. We select (E_Y, π_Y) to be the statement to the witness. See Section 2.3 for more detail on the proposed zero-knowledge proof to be used for π_Y. Hence all of $I_Y := (E_Y, \pi_Y)$ is public knowledge. We will restrict R_{SSI} to only include isogenies with degrees that are powers of 5, the size of which will be approximately $p^{1/2}$.

We begin by considering the PreSig algorithm. KeyGen and GenR have already fixed the parameters $(y, Y), \tau$, and E_A, though we do not have access to y. We will stay close to the SQISign protocol, but must somehow incorporate the statement E_Y in a verifiable way. To do so, we begin by generating the commitment, ψ, as in SQISign, but with smaller degree. Next, we will generate an extension on the commitment, ψ', which is a map whose domain is the commitment E_1, and which is determined using a hash of E_Y. Specifically, we will use the hash of E_Y as a seed to create a sequence of integers which will be used to select the kernel generators of ψ'. The codomain of ψ' serves as our new commitment, and will be used to generate the challenge, along with the message, as in SQISign. From here, the pre-signature, $\widetilde{\sigma}$, is the map from the public key E_A to the challenge curve E_2, which can be computed using the KLPT algorithm. We will also output $\tau(P_0), \tau(Q_0)$ with the presignature for use in the Adapt step later on, so the final output is $\sigma_{presig} = \big(\widetilde{\sigma}, \tau(P_0), \tau(Q_0)\big)$. Figure 2.1 outlines the PreSig algorithm.

Note that $\widetilde{\sigma}$ has too large of a degree to be considered an SIDH isogeny, which makes it impossible to use $\widetilde{\sigma}$ as-is in SIDH. However, since we will have access to all of the private information about the entire map $\widetilde{\sigma}$ during the latter portion of the protocol, we can simply break up $\widetilde{\sigma}$ into segments with SIDH-sized degrees (about $p^{1/2}$). We do so by writing $\widetilde{\sigma}$ as the composition of several isogenies with SIDH-sized degrees. We won't need to publish any auxiliary points, so this technique leaks no extra information.

Verification of the pre-signature is inherited from the verification algorithm for SQISign.

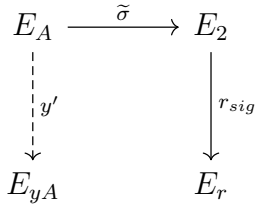

Figure 2.2: Adapt and Extract Algorithms

We now turn to the Adapt and Extract algorithms. Using the witness, y, we run an SIDH-like transformation on τ, to obtain a new map y' and its codomain E_{yA}. In other words, if $E_Y = E_0/\langle S_y \rangle$ and $E_A = E_0/\langle S_A \rangle$ then $E_{yA} = E_0/\langle S_y, S_A \rangle$. This operation requires some extra information about τ which was provided with the presignature. Now, we compute an SIDH-like square on $\widetilde{\sigma}$ and y' to obtain r_{sig}, as pictured in Figure 2.2. Note that for this second SIDH iteration, we have access to both maps y' and $\widetilde{\sigma}$ so there is no need to publish additional auxiliary points, which is why, along with the difference in degree sizes, we cannot call these explicit SIDH iterations. This completes Adapt. Note, the full signature is $\sigma = (r_{sig}, \pi_{y'})$, where $\pi_{y'}$ is a zero-knowledge proof that verifies that E_{yA} was generated honestly using y.

For Extract we can similarly compute the SIDH square for r_{sig} and $\widetilde{\sigma}$ to obtain a map from E_A to E_{yA} of appropriate degree. This gives us y', which is enough to obtain S_y, and hence y.

Both Adapt and Extract are outlined in Figure 2.2.

2.1.1 Design Choices

We give a brief summary of other designs that were considered in the construction of SQI-AS, and why they were not viable candidates.

We begin by considering the PreSig algorithm, where the output must somehow depend on the statement being used from the set R_{SSI}. Perhaps the most natural approach would be to shift the commitment curve by the statement curve using a variation of SIDH, depicted in Figure 2.3. With enough public information about the statement this could be done using B-SIDH [6]. The problem, however, is that we would end up with an unknown map from E_0 to E_Y (recall we do not have access to the witness at this point), then a known map from E_Y to E_C. This is enough to generate a challenge, but it is not enough to run the signing algorithm. In order to generate the presignature, we must run $KLPT$

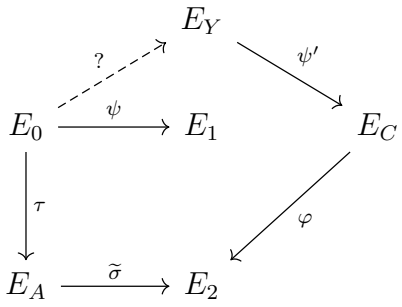

Figure 2.3: Naive approach to PreSig.

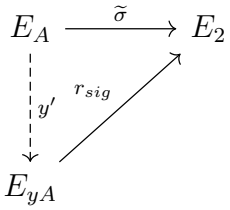

Figure 2.4: Naive approach to Adapt and Ext.

on the composition of $\hat{\tau}$, the commitment, and the challenge. If we do not know all of the maps used in the commitment, then we cannot run $KLPT$. This is why we chose, instead, to use the statement curve to canonically define an additional commitment map. When choosing this extension we considered using a multiplication by n map, where n is the j-invariant of the statement curve, however, this map would be too costly to compute. This lead us to using a hash function to obtain a canonical stream of "random" integers to choose the kernel of the extension map, as described in Section 2.1.

Now we turn to the Adapt and Extract algorithms. The intuitive approach to relating the witness y' and presignature $\widetilde{\sigma}$ maps would be to simply compose them, shown in Figure2.4. Then, when a user has the presignature and signature maps, they can compose the presignature and the signature's dual map to get back an isogeny from E_A and E_{yA}, and use KLPT to write the result as a single map. The problem that arises, however, is that the degree of an output isogeny from KLPT is fixed, and larger than required. Hence, the resulting map would not satisfy the requirements of R_{SSI}. By changing the protocol to use a variation of SIDH, we are able to recover an isogeny of the desired, fixed degree, and hence extract a viable witness.

2.2 Parameters

The main difference between standard SQISign and SQISign as used in our scheme is the degree of the commitment ψ. Other choices such as security parameters and primes will be borrowed directly from SQISign. In the SQISign [8] implementation, ψ has degree $p^2 - 1$. Since we are adding the extension ψ' we will change the degree of ψ to $p + 1$ and choose ψ' to have degree $p - 1$. This can be done efficiently through the composition of smaller prime degree isogenies, the generators of which are determined by a pseudorandom number generator. The seed that generates the sequence of numbers for ψ' will be the hash of E_Y and the seed of ψ. As discussed in [8], the commitment ψ is still secure at this degree, and so we can make ψ' public knowledge without risking any security. This means the verifier can check for themselves that ψ' was generated honestly.

As mentioned previously, R_{SSI} will be a set of isogenies with degrees of powers of 5. Following the parameters put forth in SQISign [8], we have that the presignature $\widetilde{\sigma}$ will have degree $2^l \approx p^4$. We will choose τ to correspond to an isogeny of degree $3^s \approx p^{1/2}$. This ensures that y, τ, and $\widetilde{\sigma}$ in the scheme will all have degrees that are pairwise relatively prime to each other, hence we will not run into problems during the pseudo-SIDH iterations. Recall that $\widetilde{\sigma}$ has too large a degree to be considered an SIDH isogeny, but since we will have access to the entire map during that portion of the scheme, we won't need to publish any auxiliary points, so we leak no more information.

2.3 Zero-knowledge Proof

We now give a suggestion for which sigma protocol to use for π_y in the presignature $\widetilde{\sigma}$ and in $\pi_{y'}$ for the full signature σ. In the interest of memory conservation we propose using a not-yet published zero-knowledge proof, developed at the Supersingular Isogeny Graphs in Cryptography workshop hosted at the Banff International Research Station in 2021 [1], but we also provide a zero-knowledge proof published recently [12] for use in the meantime. We begin with the former. It is a Proof of Isogeny Knowledge (PoIK) that proves knowledge of the witness in the Weak SIDH Relation, which we will recall.

2.3.1 Weak SIDH Proof of Isogeny Knowledge

We state the Weak SIDH Relation below; however, note that it is simply the CSSI Problem 1.1.2, without the degree constraint on the secret isogeny.

Problem 2.3.1. Let p be a prime of the form $p = l_A^{e_A} l_B^{e_B} f - 1$. Consider two isogenous curves E and $E/\langle G \rangle$ defined over $\mathbb{F}_{p^2}$. Assuming it exists, find an isogeny $\phi : E \to E/\langle G \rangle$ with (cyclic) kernel $\ker \phi = G$, or equivalently find a generator for G.

We summarize how one iteration of the PoIK's sigma protocol works. Depending on the requirements and desired security level, several iterations can be instantiated at the same time, and/or the protocol can be made non-interactive.

Define $B_i(E, l^e)$ to be a deterministic algorithm for computing a basis (P_i, Q_i) of $E[l^e]$. Let ϕ be an isogeny $\phi : E_0 \to E_1$, of degree $l_B^{e_B}$ such that $\phi(P_0) = P_1, \phi(Q_0) = Q_1$, where $(P_0, Q_0) = B_0(E_0, l_B^{e_B})$

Commitment: Using a random $a \in \mathbb{Z}/l_A^{e_A}\mathbb{Z}$, compute $E_2 = E_0/\langle P_0 + aQ_0 \rangle$ and $E_3 = E_1/\langle P_1 + aQ_1 \rangle$. Push ϕ forward to an isogeny $\phi' : E_2 \to E_3$. Compute $B_2(E_2, l_A^{e_A})$ and $B_3(E_3, l_A^{e_A})$. Compute a matrix

$$C = \begin{pmatrix} x & y \\ w & z \end{pmatrix}$$

such that $\phi'\big(B_2(E_2, l_A^{e_A})\big) = C \cdot B_3(E_3, l_B^{e_B})$. Output a hash of C.

Challenge: The challenger selects and sends $c \in \{0, 1\}$ at random.

Response: If $c = 0$, then output (a, C).

If $c = 1$, compute $B_*(E_2, l_B^{e_B})$. Find $e \in \mathbb{Z}/l_B^{e_B}\mathbb{Z}$ such that $\ker \phi' = \langle P^* + eQ^* \rangle$. Output (e, E_2).

Verification:

- If $c = 0$, compute $E_2 = E_0/\langle P_0 + aQ_0 \rangle$ and $E_3 = E_1/\langle P_1 + aQ_1 \rangle$. Compute $B_2(E_2, l_A^{e_A})$ and $B_3(E_3, l_A^{e_A})$.
  Compute $(P, Q)^T = C \cdot (P_3, Q_3)^T$.
  Let $\hat{\psi} : E_2 \to E_0$ and $\hat{\psi}' : E_3 \to E_1$ be the dual isogenies computed using a. Check that $\ker \psi$ can be generated by a linear combination of P_2 and Q_2, and that $\ker \psi'$ can be generated by the same linear combination using P and Q.

- If $c = 1$, compute $B_*(E_2, l_B^{e_B})$. Check that $E_3 = E_2/\langle P^* + eQ^* \rangle$. Compute $B_2(E_2, l_A^{e_A})$ and $B_3(E_3, l_A^{e_A})$. Compute $\phi(P_2)$, $\phi(Q_2)$, and recompute C.

We omitted some details for brevity. The entire proof of knowledge is estimated to be 15.5 KB for a 128 bits security level.

2.3.2 SIDH Proof of Knowledge

We now summarize a recently published PoIK by De Feo, Dobson, Galbraith, and Zobernig [12] that requires more memory than the previous one, but has an actual proof of security. The paper begins by finding a counterexample to the soundness property of the original SIDH PoIK, and concludes with their own PoIK that does satisfy the soundness property. We summarize one iteration of the sigma protocol.

The PoIK proves knowledge in the setting of Problem 1.1.3: the SIDH Relation, where we have access to the expected degrees of the isogenies, and the isogenies acting on the public basis points. Suppose the prover is proving knowledge of $\phi : E_0 \to E_1$.

Commitment: The prover will select a new isogeny $\psi : E_0 \to E_2$ at random and compute the SIDH square between E_1 and E_2 resulting in a hypothetical shared secret, E_3. See Section1.2.5 on how to compute the SIDH square. Then they will send the relevant curves E_2, E_3 to the verifier. The protocol differs from the original SIDH PoIK in that the prover will also reveal some points from E_3 in order to further commit and eliminate any opportunity of foul play on their part. This additional information is enough to achieve special soundness. To choose these points, the prover will evaluate ϕ' on two basis points for E_2, generated by a deterministic algorithm.

Challenge: The challenger selects and sends $c \in \{0, 1\}$ at random.

Response: If $c = 0$, then the prover reveals the vertical isogenies ψ, ψ'. If $c = 1$, then the prover reveals ϕ'.

Verification: In either case the verifier accepts if the revealed isogenies are of the correct degrees and correspond to the correct curves. In the latter case they will also check that the basis points of E_2 evaluate correctly.

Several iterations are necessary to achieve an adequate level of security, the details of which, along with the proofs of security, can be viewed in [12]. The sigma protocol is depicted in Figure 2.5.

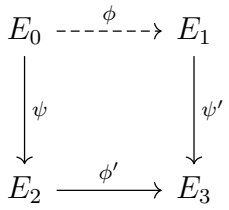

Figure 2.5: SIDH Proof of Knowledge sigma protocol.

Chapter 3

Security Proofs

We now go into the details of verifying the security of SQI-AS by checking that it satisfies the definitions outlined in Section 1.2.3.

Theorem 3.0.1. *If the signature scheme SQISign, $\Sigma_{\mathsf{SQISign}}$, is SUF-CMA, and R_{SSI} is a hard relation, then the adaptor signature scheme $\Xi_{\Sigma_{SQISign},\mathsf{R}_{SSI}}$ is secure in QROM.*

Proof. We will show that SQI-AS is secure in QROM by demonstrating it is pre-signature adaptable, pre-signature correct, aEUF-CMA, and witness extractable.

Lemma 3.0.2 (Pre-signature Adaptability)**.** *The adaptor signature scheme $\Xi_{\Sigma_{SQISign},\mathsf{R}_{SSI}}$ satisfies pre-signature adaptability.*

Proof. Fix $\big(y, I_Y := (E_Y, P_0, Q_0, \pi_Y)\big)$ to be a valid witness/statement pair from R_{SSI}. Let $\widetilde{\sigma}$ be a pre-signature generated from PreSig, with fixed parameters τ, E_A generated from KeyGen, $m \in \{0,1\}^*$, and E_Y.

Suppose we have that $\mathsf{PreVer}_{pk}(m, I_Y; \widetilde{\sigma}) = 1$. This means that $\widetilde{\sigma}$ has domain E_A and codomain E_2, where E_2 is an honestly generated challenge.

This gives us that $\mathsf{Adapt}(\widetilde{\sigma}, y)$ will necessarily be an isogeny r with domain E_2 and where its codomain is $E_A/\langle \tau(P_0) + \alpha\tau(Q_0), ker(\widetilde{\sigma})\rangle$, where $P_0 + \alpha Q_0$ is the kernel generator of y. It will also include a zero-knowledge proof that y was used to generate E_{yA}.

Now, when we apply Verify, it will successfully identify that the domain and codomain of r are as described, and that y was correctly used to generate E_{yA}. So we get that $\mathsf{Verify}_{pk}(m; \mathsf{Adapt}_{pk}(\widetilde{\sigma}, y)) = 1$. □

Lemma 3.0.3 (Pre-signature Correctness)**.** *The adaptor signature scheme* $\Xi_{\Sigma_{SQISign},\mathsf{R}_{SSI}}$ *satisfies pre-signature correctness.*

Proof. Let $\tau, E_A, y, I_Y, \widetilde{\sigma}$, and σ be as in the previous lemma.

$\mathsf{PreVer}_{pk}(m, E_Y; \widetilde{\sigma})$ will check whether the domain of $\widetilde{\sigma}$ is E_A, and its codomain is an honestly generated challenge depending on m and E_Y. Since $\mathsf{PreSig}_{sk}(m, E_Y)$ uses the KLPT algorithm to generate $\widetilde{\sigma}$ from the composition of the honestly generated challenge, commitment, and τ, by the correctness of KLPT we get that $\mathsf{PreVer}_{pk}(m, E_Y; \widetilde{\sigma}) = 1$. From the previous lemma, we thus get that $\mathsf{Verify}_{pk}(m; \mathsf{Adapt}_{pk}(\widetilde{\sigma}, y)) = 1$ as well.

Now it remains to show that $y^* := \mathsf{Ext}_{pk}(\sigma, \widetilde{\sigma}, I_Y)$ is an isogeny from the base curve E_0 to E_Y, and thus that $(y^*, I_Y) \in \mathsf{R}_{SSI}$. In $\mathsf{Ext}_{pk}(\sigma, \widetilde{\sigma}, I_Y)$ the maps $\hat{\sigma}$ and r are used to compute an SIDH square in order to obtain an isogeny from E_A to E_{yA}. We call this isogeny y'. Now we may compute the kernel generator of y' as a linear combination of $\tau(P_Y), \tau(Q_Y)$, giving the secret integer α^*, where α^* is such that $\tau(P_Y) + \alpha^*\tau(Q_Y)$ is the kernel generator of y'. Note that the values $\tau(P_Y), \tau(Q_Y)$ are shared as part of the pre-signature $\widetilde{\sigma}$.

We define y^* to be the isogeny with codomain E_0 whose kernel generator is $P_Y + \alpha^* Q_Y$. By the correctness of SIDH, we are assured that the α used to generate y is the same as α^*, hence $(y^*, I_Y) \in \mathsf{R}_{SSI}$. □

Lemma 3.0.4 (aEUF-CMA Security)**.** *If the signature scheme SQISign,* $\Sigma_{\mathsf{SQISign}}$ *is SUF-CMA secure, and* R_{SSI}*, is a hard relation, then the adaptor signature scheme* $\Xi_{\Sigma_{SQISign},\mathsf{R}_{SSI}}$ *is aEUF-CMA secure.*

In terms of the structure of our protocol and the hard problems being used in it, ours is most similar to IAS, the adaptor signature described in [23]. In their proof of aEUF-CMA security, the authors focus on detailing how an adversary can use the CSI-FiSh signing oracle to answer both signing and pre-signing queries. They have the adversary use the CSI-FiSh oracle to answer signing queries, and when a pre-signing query is made they again use the oracle to make a full signature, obtain the witness of the relevant statement, and make the pre-signature from these parts.

The case of SQI-AS is different, in that we may only use our underlying signature–SQISign–to answer pre-signature query calls. This is due to the fact that our final signature is of a necessarily different degree than the output of a SQISign signature. Hence, we put forth a proof which reduces the security of SQI-AS to the security of SQISign, with some changes to the proof structure in [23].

We begin with our proof of reducing to the security of SQISign, where the key difference is that our signing oracle will require a statement as input in order to answer pre-signing

$\mathsf{aSigForge}_{\mathcal{A},\mathsf{aSIG}_{\mathsf{R},\mathsf{SIG}}}(n)$	$\mathcal{O}_S(m,Y)$	$\mathcal{O}_{pS}(m,Y)$
$1:\mathcal{Q}:=\emptyset,(sk,pk)\leftarrow\mathsf{Gen}(1^n)$	$1:\sigma\leftarrow\mathsf{Sign}_{sk}(m,Y)$	$1:\widetilde{\sigma}\leftarrow\mathsf{pSign}_{sk}(m,Y)$
$2:m^*\leftarrow\mathcal{A}^{\mathcal{O}_S,\mathcal{O}_{pS}}(pk)$	$2:\mathcal{Q}:=\mathcal{Q}\cup\{m\}$	$2:\mathcal{Q}:=\mathcal{Q}\cup\{m\}$
$3:(Y,y)\leftarrow\mathsf{GenR}(1^n),\widetilde{\sigma}\leftarrow\mathsf{pSign}_{sk}(m^*,Y)$	$3:$ **return** σ	$3:$ **return** $\widetilde{\sigma}$
$4:\sigma^*\leftarrow\mathcal{A}^{\mathcal{O}_S,\mathcal{O}_{pS}}(\widetilde{\sigma},Y)$		
$5:$ **return** $\left(m^*\notin\mathcal{Q}\wedge\mathsf{Vrfy}_{pk}(m^*;\sigma^*)\right)$		

Figure 3.1: The modified aSigForge game.

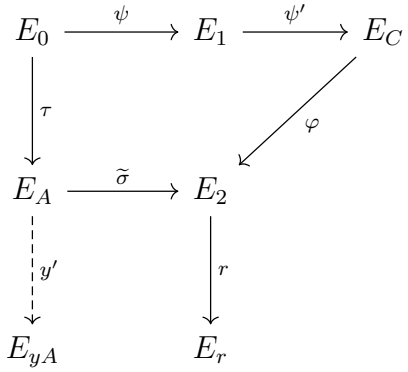

Figure 3.2: SQI-AS.

queries. This differs from Definition 1.2.3, but seeing as the statement is generally public knowledge, and available to all parties involved in adaptor signature transactions, we claim this would not harm the work-flow of the protocol substantially. In the interest of clarity, we provide an amended version of the game used in Definition 1.2.3 in Figure 3.1 to be referred to in our proof, along with a diagram of SQI-AS in Figure 3.2 for ease of presentation.

Proof. We proceed by reducing the forgeability of $\Xi_{\Sigma_{SQISign},\mathsf{R}_{SSI}}$ (SQI-AS) to that of the signature scheme SQISign. We play the amended aSigForge game with an adversary $\mathcal{A}$, and use their forgery to win the forgeability experiment of SQISign. We first focus on how to supply the signing and pre-signing queries to $\mathcal{A}$. Assuming we can answer these calls, then we will be able to use their forgery to win our aSigForge game.

We begin by considering a sequence of games, starting with the aSigForge game, where we will be able to answer all query calls made by $\mathcal{A}$ by the last game in the sequence.

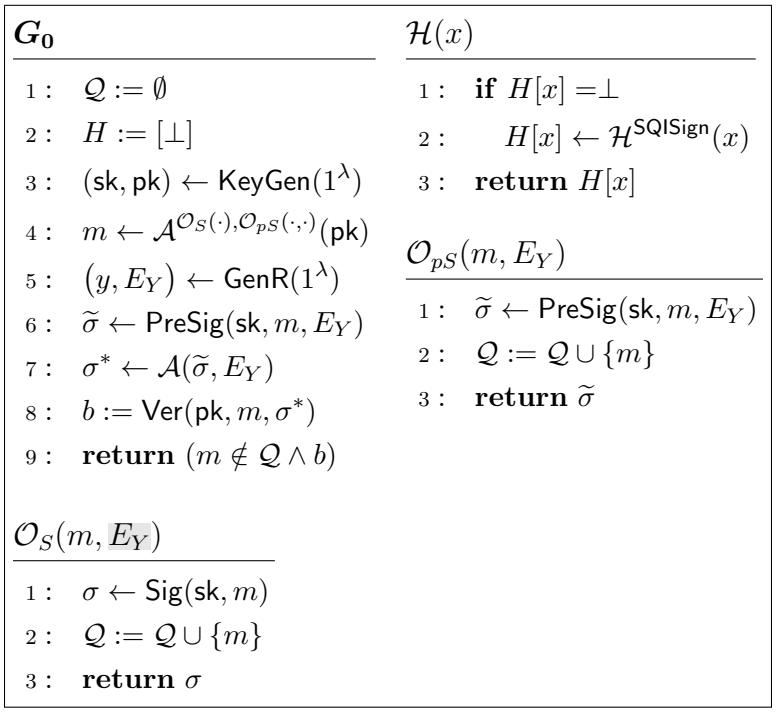

Figure 3.3: Game 0–aSigForge

We include a summary of each game and how it changes in the progression, along with the games written in pseudocode. We closely follow the structure of the equivalent proof in [23], but differ in our modifications of the signing oracle instead of the pre-signing oracle (starting in game $\boldsymbol{G_2}$).

Game $\boldsymbol{G_0}$: This game is the aSigForge game 3.1. $\mathcal{A}$ can query the pre-signing oracle $\mathcal{O}_{pS}$ and the signing oracle $\mathcal{O}_S$. Since we are in the random oracle model, $\mathcal{A}$ can also query $\mathcal{H}$, the random oracle. The game corresponds to the adversary $\mathcal{A}$ creating a forgery σ^* for the message m^* of its choosing. Because $\boldsymbol{G_0}$ is exactly the aSigForge game we get that

$$Pr[\mathsf{aSigForge}_{\mathcal{A},\Xi}(n) = 1] = Pr[\boldsymbol{G_0} = 1]$$

Game $\boldsymbol{G_1}$: This game is the same as $\boldsymbol{G_0}$ except that once a forgery σ^* is outputted, a win condition is added to the game in step 8 of Figure 3.4 (or in Step 5 of Figure 3.1) where it checks if σ^* can be obtained by adapting the pre-signature $\widetilde{\sigma}$ with the witness y. This can be done since the game itself generates the key pairings in Step 1 and 3 of Figure 3.4.

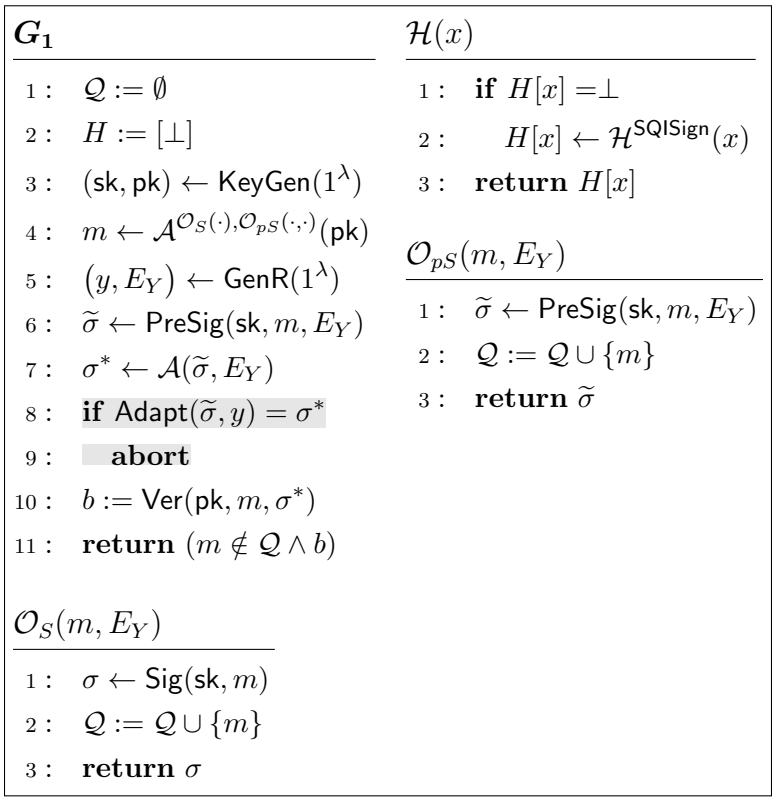

Figure 3.4: Game 1

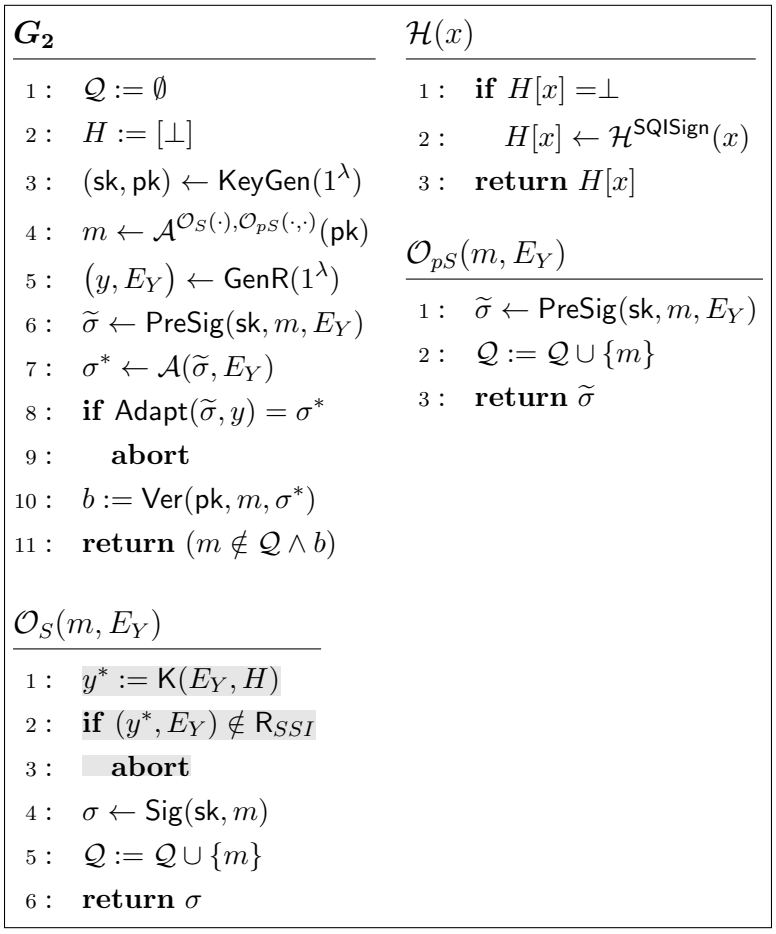

Figure 3.5: Game 2

If it can be adapted in this way, then the game aborts. Note, while the witness/statement pair is generated within the game, only the statement is made public to $\mathcal{A}$.

We claim that the difference in win probability between $\boldsymbol{G_1}$ and $\boldsymbol{G_0}$ is negligible. If it were non-negligible, we would be able to use the statement and forgery, along with the pre-signing oracle, to extract a valid witness, thereby breaking the security of our hard relation R_{SSI}. Hence,

$$Pr[\boldsymbol{G_1} = 1] \leq Pr[\boldsymbol{G_0} = 1] + \mathsf{negl}(\lambda)$$

Game $\boldsymbol{G_2}$: Here is where we will make use of the input of the statement curve E_Y to the signing oracle. In this game the only difference from $\boldsymbol{G_1}$ is that we modify the signing

oracle $\mathcal{O}_S$. During the signing queries, we use the QROM extractor algorithm K on input of the statement curve E_Y and the list of random oracle queries H to extract a witness y^*. Specifically we do so as follows: recall that in the Quantum Random Oracle Model, since the proof of knowledge of the witness requires the prover to submit hashes of the responses to all possible challenges, the oracle will be able to extract the witness using its online extractor algorithm K. For more details on this process see Section 1.2.6 or [24]. In the case that $(y^*, E_Y) \notin R_{SSI}$ the game aborts. Hence, the game now has access to the relevant witness, for use in the signing oracles.

We claim that the probability of $\boldsymbol{G_2}$ aborting due to extracting a witness y^* such that $(y^*, E_Y) \notin R_{SSI}$ is negligible. This is due to the *extractor property* of the zero-knowledge proof being submitted to the random oracle. Since multiple proofs of knowledge of the witness must be submitted, there is a non-negligible probability that a substantial subset of them must be correct. Since the oracle requires only one fully correct proof to meet the online extractablility algorithm, it is expected that K will be able to recover a viable witness to the given statement curve. Since $\boldsymbol{G_1}$ and $\boldsymbol{G_2}$ differ only in the previous abort event, we get that

$$Pr[\boldsymbol{G_2} = 1] \leq Pr[\boldsymbol{G_1} = 1] + \mathsf{negl}(\lambda)$$

Game $\boldsymbol{G_3}$: In this game we further modify the signing algorithm $\mathcal{O}_S$. In $\mathcal{O}_S$ we will execute the PreSig algorithm to obtain a correct pre-signature $\widetilde{\sigma}$, and then converting it into a valid signature using the extracted witness y that we obtained from K. The game must also simulate the proofs included in the signature. Because of the zero-knowledge property of our zero-knowledge proof, the simulator is able to generate a proof indistinguishable from an honest one, π^*.

We use the regular Adapt algorithm to obtain the signature, so this game is indistinguishable from the previous game. It holds that

$$Pr[\boldsymbol{G_3} = 1] \leq Pr[\boldsymbol{G_2} = 1] + \mathsf{negl}(\lambda)$$

Hence, the original $\mathsf{aSigForge}$ game is indistinguishable to $\boldsymbol{G_3}$, a game we can answer the queries for.

Now we have reduced the original $\mathsf{aSigForge}$ game to $\boldsymbol{G_3}$, a game where we can answer the query calls of $\mathcal{A}$. Specifically, when $\mathcal{A}$ queries the pre-signing oracle, the simulator will query the SQISign signing oracle. When $\mathcal{A}$ queries the signing oracle, the simulator first queries the SQISign signing oracle, uses the extractor K, and then uses the resulting witness to adapt the pre-signature into a valid signature. Now $\mathcal{A}$ can make any queries to

G_3

1 : $\mathcal{Q} := \emptyset$
2 : $H := [\perp]$
3 : $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$
4 : $m \leftarrow \mathcal{A}^{\mathcal{O}_S(\cdot), \mathcal{O}_{pS}(\cdot,\cdot)}(\mathsf{pk})$
5 : $(y, E_Y) \leftarrow \mathsf{GenR}(1^\lambda)$
6 : $\widetilde{\sigma} \leftarrow \mathsf{PreSig}(\mathsf{sk}, m, E_Y)$
7 : $\sigma^* \leftarrow \mathcal{A}(\widetilde{\sigma}, E_Y)$
8 : **if** $\mathsf{Adapt}(\widetilde{\sigma}, y) = \sigma^*$
9 : **abort**
10 : $b := \mathsf{Ver}(\mathsf{pk}, m, \sigma^*)$
11 : **return** $(m \notin \mathcal{Q} \wedge b)$

$\mathcal{O}_S(m, E_Y)$

1 : $y^* := \mathsf{K}(E_Y, H)$
2 : **if** $(y^*, E_Y) \notin \mathsf{R}_{SSI}$
3 : **abort**
4 : $\widetilde{\sigma} \leftarrow \mathsf{PreSig}(\mathsf{sk}, m, E_Y)$
5 : $\sigma \leftarrow \mathsf{Adapt}(\widetilde{\sigma}, y, (P_Y, Q_Y))$
6 : $\mathcal{Q} := \mathcal{Q} \cup \{m\}$
7 : **return** σ

$\mathcal{H}(x)$

1 : **if** $H[x] = \perp$
2 : $H[x] \leftarrow \mathcal{H}^{\mathsf{SQISign}}(x)$
3 : **return** $H[x]$

$\mathcal{O}_{pS}(m, E_Y)$

1 : $\widetilde{\sigma} \leftarrow \mathsf{PreSig}(\mathsf{sk}, m, E_Y)$
2 : $\mathcal{Q} := \mathcal{Q} \cup \{m\}$
3 : **return** $\widetilde{\sigma}$

Figure 3.6: Game 3

the oracles it requires, and is able to generate a forgery. It remains to show that we can use the resulting forgery provided by $\mathcal{A}$ to win the SQISign $\mathsf{SigForge}$ game.

To start the attack on SQISign, after $\mathcal{A}$ has made its chosen $\mathcal{O}_S$ and $\mathcal{O}_{pS}$ query calls they will eventually generate their chosen message m^*. The simulator seeking to forge a SQISign signature will use m^* as the challenge message in the SQISign attack. Now, in order to forge a signature on the challenge message m^*, the simulator will consider the challenge isogeny generated in SQISign for the message m^*, and give this map to $\mathcal{A}$ as the pre-signature, along with E_0. The simulator will simulate a zero-knowledge proof that E_0 was generated using a witness y where $(y, E_0) \in \mathsf{R}_{SSI}$. This is possible due to the zero-knowledge property of our proof. Hence, the forgery σ^* provided by $\mathcal{A}$ will be an isogeny from E_A to the challenge curve, as required by SQISign. This isogeny will not be of the correct degree size (it will have a smaller degree), so we run an iteration of KLPT on it to achieve the desired degree. This will be sufficient to satisfy the SQISign Verify algorithm. □

Lemma 3.0.5 (Witness Extractability)**.** *If the signature scheme SQISign,* $\Sigma_{\mathsf{SQISign}}$ *is* $\mathsf{SUF-CMA}$*, and* R_{SSI}*, is a hard relation, then the adaptor signature scheme* $\Xi_{\Sigma_{SQISign},\mathsf{R}_{SSI}}$ *is witness extractable.*

Proof. The difference now from the previous lemma is that $\mathcal{A}$ outputs I_Y along with the message m^*. The simulator $\mathcal{S}$ does not have access to y, so it cannot use the witness to adapt the pre-signatures into full signatures. However, it is possible to extract y from the zero-knowledge proof included in the statement I_Y since we are in the QROM setting.

Thus the proof is almost identical to that of Lemma 3.0.4 except for the detail on obtaining y described above.

□

This concludes the proof of Theorem 3.0.1. □

Chapter 4

Use on Blockchain

4.1 Payment Channel Networks

The main functionality of adaptor signatures is for use on blockchains. In the case two parties will be sending several transactions to each other, the on-chain cost can get very expensive. To avoid these costs, the two parties can use a *payment channel network* (PCN) at the cost of two on-chain transactions. This construction was introduced in [3], where the full details are included. We give a brief description of the primitive in order to motivate the work put into SQI-AS.

We suppose Alice and Bob would like to exchange cryptocurrencies at least three times. They will first agree on a starting balance, each one starting with T_A and T_B coins respectively. They submit these amounts for authentication on the blockchain, which essentially "opens" the channel between them, locking the coins into the wallet. Now let's say Alice wants to send Bob $t \leq T_A$ coins. Then both parties will sign the new balances of $T_A - t$ and $T_B + t$ respectively, and keep the confirmations locally. They can repeat this step any number of times. Once they decide they would like to close the channel, they will submit the most recent channel balance for authentication on the blockchain, and are now able to cash out.

We run into the problem that Alice and Bob might not have a direct line of communication already open between them, for which we introduce the notion of an *anonymous multi-hop lock* (AMHL). Alice and Bob can identify a chain of communication through some number of intermediaries $\{U_i\}, i = 1, \ldots, k$. Each intermediary U_i will establish a lock on the right with U_{i+1}, in which they agree to release funds to U_{i+1} if they can solve

a cryptographic problem. We first outline the construction from [23], and then highlight how it can be changed for use with SQI-AS.

4.1.1 AMHL realized by IAS

There are three phases: set-up, commit, and release. We have a sender S sending funds to a receiver R through the intermediaries $U_1, \ldots, U_k$. In the set-up phase, S will choose some random strings $\{l_i\}_{i=1}^{k}$, one for each intermediary. Next, S will compute $y_j = \sum_{i=0}^{j} l_i$ and $Y_j = f(y_j)$, where f is the cryptographic function being used (in the case of [23] it is the ideal class group action on supersingular elliptic curves). S will now send the tuple (Y_i, Y_{i-1}, l_i) to each intermediary U_i, and sends (Y_k, y_k) to R. Now S must also send a zero-knowledge proof to each U_i that S in fact knows a witness to their given statement.

For the commit phase, each intermediary will create a pre-signature $\hat{\sigma}_i = \mathsf{PreSig}(\mathsf{sk_i}, \mathsf{tx_i}, \mathsf{Y_i})$ where $\mathsf{tx_i}$ is the conditional contract stating that U_i will release funds to U_{i+1} once U_i is provided their full signature. Beginning in the release phase, R has enough information to immediately provide U_k with their full signature–that is, since R has access to y_k and $\hat{\sigma}_k$, R can immediately compute σ_k. Once U_i has received σ_i, they can use their own pre-signature $\hat{\sigma}_i$ to extract their witness y_i. From here, using l_i, they compute $y_{i-1} = y_i - l_i$, use this to adapt $\hat{\sigma}_{i-1}$ and sends σ_{i-1} to U_{i-1}, thereby releasing their funds. This process will repeat until S has released their own funds.

4.1.2 AMHL realized by SQI-AS

The same idea can be applied to SQI-AS, with a slight change in the set-up phase. Since our witnesses are not directly integers, we instead use the integers to compute the secret isogeny. We repeat the set-up phase for clarity.

S begins by selecting a random integer $l_j \in \mathbb{F}_{p^2}$ for each U_j, and then computes $\alpha_j = \sum_{i=0}^{j} l_i$. Since in each instance of R_{SSI} the statement includes a basis $\{P_Y, Q_Y\}$ that can be canonically computed, we choose our witnesses $\{y_i\}$ such that $ker(y_j) = \langle P_Y + \alpha_j Q_Y \rangle$, giving us that $Y_j = codomain(y_j)$. The rest of the scheme is essentially the same, with the added computation that U_i will have to compute a discrete logarithm to obtain α_i from their witness y_i.

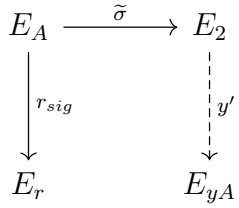

Figure 4.1: New Adapt and Extract algorithms for use on-chain.

4.2 On-Chain Use

Using adaptor signatures in payment-channel networks will also involve full signatures on the block chain. For this reason, we would require our adapted signature to look just like a regular signature. Hence, we outline a variant of SQI-AS that achieves this but at a cost to efficiency.

The only difference comes from the order of the SIDH operations in the Adapt and Extract algorithms.

In this variant, we perform the SIDH operation on $\widetilde{\sigma} \circ \tau$ instead of just τ. Specifically, in the presignature we will publish $\widetilde{\sigma} \circ \tau(P_Y)$ and $\widetilde{\sigma} \circ \tau(Q_Y)$. Then in Adapt, once we have access to the witness y, and hence its kernel generator $P_Y + \alpha Q_Y$, we can use α to compute $\widetilde{\sigma} \circ \tau(P_Y) + \alpha \cdot \widetilde{\sigma} \circ \tau(Q_Y)$, which we will use as the kernel generator for y'. Lastly, we compute the SIDH square on y' and $\widetilde{\sigma}$ to obtain our full signature r_{sig}, as pictured in Figure 4.1.

The Extract algorithm is completed by computing the SIDH square on $\widetilde{\sigma}$ and r_{sig}, and from there extracting α.

In this variant, the adapted signatures satisfy the SQISign verification algorithm, using only a new challenge (instead of a different public key, as seen in the previous version). Hence, the verifier need only check that the extension on the challenge was honestly generated. The efficiency loss comes from this check and having to run the additional SIDH operation to compute y'. The overall loss is not significant, and so this variant is still feasible to use.

Chapter 5

Cost Estimates and Conclusion

We suggest following the SQISign parameter guidelines [8], with the slight changes made to the commitment degree size as outlined in Section 2.2. Specifically, the authors of SQISign suggest using the prime p such that

$$p+1 = 2^{33}\cdot 5^{21}\cdot 7^2\cdot 11\cdot 31\cdot 83\cdot 107\cdot 137\cdot 751\cdot 827\cdot 3691\cdot 4019\cdot 6983\cdot 517434778561\cdot 26602537156291,$$

$$p-1 = 2\cdot 3^{53}\cdot 43\cdot 103^2\cdot 109\cdot 199\cdot 227\cdot 419\cdot 491\cdot 569\cdot 631\cdot 677\cdot 857\cdot 859\cdot 883\cdot 1019\cdot 1171\cdot 1879\cdot 2713\cdot 4283.$$

We estimate the storage costs of the pre-signature, adapted signature, and key generation, and compare the results to IAS [23], an isogeny-based adaptor signature with CSI-FiSh used as its underlying signature scheme. Our estimated pre-signature requires only 226 bytes, instead of more than 18 000 bytes as in IAS. Unfortunately, our signature requires a zero-knowledge proof as well, so its size would be about 15 704 bytes, where as the IAS signature can range between 263 and 1880 depending on the security parameters used. Thus, SQI-AS provides a significant improvement in pre-signature size, but at the cost of a larger adapted signature. This could be optimal if the payment channel network is very large and requires a long set-up.

Future work can additionally look to improve the size and security of the zero-knowledge proof being used. A shorter proof would have a large effect on the size of the adapted signature, which is significant when signing on the blockchain. A concrete implementation is also left to future work.

References

[1] Weak SIDH proof of isogeny knowledge. In *Proceedings of the Supersingular Isogeny Graphs in Cryptography workshop (online) at Banff International Research Station for Mathematical Innovation and Discovery*, pages 15–18, 2021.

[2] 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. *Selected Areas in Cryptography – SAC 2018*, pages 322–343, 2019.

[3] Lukas Aumayr, Oguzhan Ersoy, Andreas Erwig, Sebastian Faust, Kristina Hostáková, Matteo Maffei, Pedro A. Moreno-Sanchez, and Siavash Riahi. Generalized bitcoin-compatible channels. *IACR-ASIACRYPT-2021*, page 476, 2020.

[4] Ward Beullens, Thorsten Kleinjung, and Frederik Vercauteren. CSI-FiSh: Efficient isogeny based signatures through class group computations. *Advances in Cryptology – ASIACRYPT 2019*, pages 227–247, 2019.

[5] Wouter Castryck, Tanja Lange, Chloe Martindale, Lorenz Panny, and Joost Renes. CSIDH: An efficient post-quantum commutative group action. *Advances in Cryptology – ASIACRYPT 2018*, 2018.

[6] Craig Costello. B-SIDH: supersingular isogeny Diffie-Hellman using twisted torsion. *Advances in Cryptology – ASIACRYPT 2020*, 2020.

[7] Luca De Feo and Steven D. Galbraith. SeaSign: Compact isogeny signatures from class group actions. *Advances in Cryptology – EUROCRYPT 2019*, pages 759–789, 2019.

[8] Luca De Feo, David Kohel, Antonin Leroux, Christophe Petit, and Benjamin Wesolowski. SQISign: compact post-quantum signatures from quaternions and isogenies. *ASIACRYPT 2020*, 2020.

[9] Leo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, Peter Schwabe, Gregor Seiler, and Damien Stehlé. CRYSTALS-Dilithium: A lattice-based digital signature scheme. *Transactions on Cryptographic Hardware and Embedded Systems*, 2018, 01 2018.

[10] Andreas Erwig, Sebastian Faust, Kristina Hostáková, Monosij Maitra, and Siavash Riahi. Two-party adaptor signatures from identification schemes. In Juan A. Garay, editor, *Public-Key Cryptography - PKC 2021*, volume 12710, pages 451 – 480, Cham, 2021. Springer.

[11] Muhammed Esgin, Oğuzhan Ersoy, and Erkin Zekeriya. Post-quantum adaptor signatures and payment channel networks. *Computer Security – ESORICS 2020 - 25th European Symposium on Research in Computer Security*, pages 378–397, 09 2020.

[12] Luca De Feo, Samuel Dobson, Steven D. Galbraith, and Lukas Zobernig. Sidh proof of knowledge. Cryptology ePrint Archive, Report 2021/1023, 2021. https://ia.cr/2021/1023.

[13] Luca De Feo, Antonin Leroux, and Benjamin Wesolowski. New algorithms for the Deuring correspondence: SQISign twice as fast. Cryptology ePrint Archive, Report 2022/234, 2022. https://ia.cr/2022/234.

[14] 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.

[15] David Jao and Vladimir Soukharev. Isogeny-based quantum-resistant undeniable signatures. In *Post-Quantum Cryptography*, pages 160–179. Springer International Publishing, Cham, 2014.

[16] 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.

[17] Giulio Malavolta, Pedro Moreno-Sanchez, Clara Schneidewind, Aniket Kate, and Matteo Maffei. Anonymous multi-hop locks for blockchain scalability and interoperability. *Network and Distributed System Security (NDSS) Symposium 2019*, 01 2019.

[18] Michele Mosca and Marco Piani. Quantum threat timeline report 2020. evolutionQ, 2021. https://www.evolutionq.com/quantum-threat-timeline-2020.html.

[19] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system, 2009. http://www.bitcoin.org/bitcoin.pdf.

[20] Victoria Quehen, Péter Kutas, Chris Leonardi, Chloe Martindale, Lorenz Panny, Christophe Petit, and Katherine Stange. Improved torsion-point attacks on SIDH variants. *Advances in Cryptology - CRYPTO 2021*, pages 432–470, 08 2021.

[21] Peter W. Shor. Algorithms for quantum computation: discrete logarithms and factoring. In *Proceedings 35th Annual Symposium on Foundations of Computer Science*, pages 124–134, 1994.

[22] Joseph H. Silverman. *The Arithmetic of Elliptic Curves*. Springer-Verlag, New York, NY, 2. aufl. edition, 2009.

[23] Erkan Tairi, Pedro Moreno-Sanchez, and Matteo Maffei. Post-quantum adaptor signature for privacy-preserving off-chain payments. In Nikita Borisov and Claudia Diaz, editors, *Financial Cryptography and Data Security*, pages 131–150. Springer Berlin Heidelberg, 2021.

[24] Dominique Unruh. Non-interactive zero-knowledge proofs in the quantum random oracle model. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology - EUROCRYPT 2015*, pages 755–784. Springer Berlin Heidelberg, 2015.