Efficient Isogeny Proofs Using Generic Techniques

Kelong Cong[1], Yi-Fu Lai[2], Shai Levin[2]

[1] imec-COSIC, KU Leuven, Leuven, Belgium
[2] University of Auckland, Auckland, New Zealand
kelong.cong@esat.kuleuven.be,
ylai276@aucklanduni.ac.nz,
shai.levin@auckland.ac.nz

Abstract. Generating supersingular elliptic curves of unknown endomorphism ring has been a problem vexing isogeny-based cryptographers for several years. A recent development has proposed a trusted setup protocol to generate such a curve, where each participant generates and proves knowledge of an isogeny. Thus, the construction of efficient proofs of knowledge of isogeny has developed new interest.

Historically, the isogeny community has assumed that obtaining isogeny proofs of knowledge from generic proof systems, such as zkSNARKs, was not a practical approach. We contribute the first concrete result in this area by applying Aurora (EUROCRYPT'19), Ligero (CCS'17) and Limbo (CCS'21) to an isogeny path relation, and comparing their performance to a state-of-the-art, tailor-made protocol for the same relation. In doing so, we show that modern generic proof systems are competitive when applied to isogeny assumptions, and provide an order of magnitude (3-10×) improvement to proof and verification times, with similar proof sizes. In addition, these proofs provide a stronger notion of soundness, and statistical zero-knowledge; a property that has only recently been achieved in isogeny PoKs. Independently, this technique shows promise as a component in the design of future isogeny-based or other post-quantum protocols.

Keywords: Isogeny, Zero-knowledge, zkSNARK, Interactive Oracle Proof, MPC-in-the-Head

1 Introduction

Isogeny-based cryptography was first introduced with the CGL hash function [CLG09] by Charles, Goren and Lauter, where the core hardness assumption is that, given two isogenous elliptic curves, it is hard to recover an isogeny between them. Several other isogeny-based protocols were proposed, including SIDH [JD11], which relaxes the assumption by giving additional torsion point information; CSIDH based on group actions [CLM$^+$18, Onu21]; SQI-Sign [DKL$^+$20]; and pSIDH. [Ler21]. Even though there was a recent cryptanalysis breakthrough on SIDH [CD22, Rob22], other cryptosystems (not based on SIDH) remain unaffected, such as [CLM$^+$18, DKL$^+$20, Ler21]. Additionally, a variety of advanced schemes and protocols based on isogenies, such as oblivious transfer and exotic signatures, have been proposed in the literature [BKV19, BKP20, LGd21, BD21, BDK$^+$22].

In every isogeny-based cryptosystem, isogeny walks start from a public curve. In the literature, the candidate is usually one of the j-invariants 0 or 1728 with a known endomorphism ring. In isogeny-based constructions, sampling an isogeny without knowing its endomorphism ring [BBD$^+$22, MMP22], is a notorious bottleneck, and is essential in some constructions and applications [CLG09, LGd21, BD21, AEK$^+$22, Ste22]. From a cryptanalytical perspective, having a public curve with an unknown endomorphism ring significantly reduces the information an attacker/analyst may have. A recent proposal [BCC$^+$22] suggests a trusted setup ceremony to resolve this problem. In the ceremony, every party computes an isogeny path from the previous curve to another, produces a proof that the isogeny was generated honestly, and disposes of the path. They then publish their new curve and associated proof publicly, which all parties verify. Once every participant has completed their round, the ceremony outputs the final curve. As long as at least one party behaves honestly, recovering the final curve's endomorphism ring is difficult, even if the rest of the participants collude.

However, generating a zero-knowledge proof of an isogeny path is not a trivial task in general. In the realm of group actions, it is not difficult to achieve and the proofs for more sophisticated relations can be made [BKV19, BKP20, BDK$^+$22, ABCP22]. However, out of realm of the group actions, the task has been known to be difficult to achieve either soundness (for the exact relation) or (statistical) zero-knowledge, with some protocols requiring ad-hoc security assumptions. The state-of-the-art line of work is given in [DFJP14, GKPV21, DDGZ21, BCC$^+$22], yet there is still room for improvement. Suppose 300 participants run the ceremony single-threaded on a normal machine, the protocol will take roughly an hour to complete for $\lambda = 128$, and 13 hours for $\lambda = 256$.

Historically, it was assumed that tailor-made proof systems for isogeny relations performed better than generic ones. However, the developments of generic proof systems, such as zkSNARKs[3], which allow a prover to prove or argue the knowledge of any NP relation, have advanced the field significantly in recent years. zkSNARKs enable a prover to produce a publicly-verifiable proof in a zero-knowledge and non-interactive manner. Moreover, the proof size is *succinct*, sublinear in the size of the witness, and the verification time is much shorter than producing the proof. The area of zero-knowledge proof systems has been very active [IKOS09, BCC$^+$16, AHIV17, KKW18, BCR$^+$19, BFH$^+$20, dOT21] (see [Tha20, Ish20] for surveys). These generic proof systems work well with symmetric primitives and have applications in post-quantum cryptosystems [ZCD$^+$20, GMNO18, dDOS19, BdK$^+$21, FJR22, FMRV22], and privacy-preserving blockchain protocols such as [BCG$^+$14].

Applying generic proof systems to isogeny-based cryptography remains uncommon. Though there exists a verifiable delay function from isogenies using

[3] zero-knowledge, succinct, non-interactive, arguments of knowledge

a SNARG[4], it is not in zero-knowledge, and the result remains theoretical in nature, with unclear practicality. In particular, due to the complexity of computing isogenies, size and the structure of the operating field, using generic proof systems in isogeny-based cryptography appears challenging and impractical. Generic proof systems have been applied to protocols utilising fields of bit length at most 256-bits, whereas many isogeny-based protocols utilise field extensions of a field of upwards of 512-bits. Due to these factors, it was previously assumed these proof systems did not scale well with isogeny-based protocols. In the isogeny community, the plausibility of the following question was largely disputed:

Can generic proof systems serve as a practical tool in isogeny-based cryptography?

1.1 Contribution

We affirm the question above. That is, generic proof systems are remarkably efficient for isogeny-based cryptography. Specifically, our contributions are:

- We propose a non-interactive protocol to prove knowledge of an isogeny path using a generic zkSNARK proof system for R1CS (rank-1 constraint systems). We achieve this by re-writing the isogeny path relation into a compact R1CS representation and then applying existing (plausibly) post-quantum proof systems [BCR$^+$19, dOT21, AHIV17]. The PoK inherits the properties of soundness and statistical zero-knowledge from the underlying proof systems, and supports supersingular isogeny graphs operating over any cryptographically sized prime of the form $p = 2^a 3^b f \pm 1$, with isogeny paths of arbitrary length.
- We provide an alternative set of parameters of the form $p = 2^a 3^b f + 1$ with equivalent security to those from SIKE to aid in our testing. These parameters are designed to better support the requirements of the underlying proof systems.
- Our protocol is implemented as a proof of concept, and we report benchmark results for a variety of parameters. Using our R1CS instances from above, the generic proof systems yield competitive results as isogeny identification schemes. In particular, by utilising Aurora [BCR$^+$19] our proof systems are 3-12 times faster than the state-of-the-art [BCC$^+$22] of the same walk length, while maintaining a similar proof size.

1.2 Related Work

The motivation behind this work is to construct an efficient isogeny proof of knowledge. One such application of which is a multi-party setup protocol to generate a supersingular curve of unknown endomorphism ring, introduced in [BCC$^+$22].

[4] succinct, non-interactive argument

We give a brief history of prior works on proving isogeny knowledge, which differs from our approach.

Prior to this work, isogeny proofs of knowledge have existed in different forms, notably [DJP11, DDGZ21]. These are Σ-protocols, tailored to the specific nature of isogeny computation, and follow a direction to the original DJP identification protocol. These protocols can be viewed as revealing different edges on the SIDH square (Fig. 1) in order to prove knowledge of the isogeny $\phi : E_0 \to E_1$ of degree $\ell_A^{e_A}$. To generate the square, the prover computes an isogeny $\psi : E_0 \to E_2$ of degree $\ell_B^{e_B}$. The prover then determines the isogenies ϕ' and ψ' by their kernels, such that $\ker \phi' = \psi(\ker \phi)$ and $\ker \psi' = \phi(\ker \psi)$.

$$\begin{tikzcd} E_0 \arrow[r, "\phi"] \arrow[d, "\psi"] & E_1 \arrow[d, "\psi'"] \\ E_2 \arrow[r, "\phi'"] & E_3 \end{tikzcd} \tag{1}$$

Fig. 1. The SIDH square

In the original De Feo-Jao-Plût identification protocol, in each iteration of the Σ-protocol, the prover generates a new SIDH square in the manner described above (with a fresh choice of ψ). The prover reveals the curves E_2, E_3. The verifier then sends a binary challenge b. If $b = 0$, the prover sends the vertical isogenies to the verifier, who checks if they are indeed isogenies of correct degree, domain, and codomain. Likewise, if $b = 1$, the prover sends the horizontal isogeny ϕ', and the prover verifies the isogeny is of correct degree and domain/codomain.

However, this protocol suffered from various issues. Aside from an ad-hoc security assumption, it did not achieve statistical zero-knowledge (since the side ϕ' is strongly correlated to the side ϕ), and possessed issues with its proof of soundness (see [DDGZ21, GPV21]). De Feo et al. 's protocol increases the challenge space to 3, and proposed a solution to soundness by including a commitment to $\ell_B^{e_B}$-torsion bases of E_2 and E_3, such that the latter is the image of the former under ϕ'.

The latest work, Secuer PoK [BCC^{+}22], resolves the problem of statistical zero-knowledge (and forgoing the need for additional assumptions) by extending the degree of ϕ and ψ by composing isogenies and gluing SIDH squares together such that the walk ψ causes uniform mixing in a particular lift of the supersingular isogeny graph[5], causing the distribution of (E_1, ϕ') to be statistically close to uniform. In addition, also they extend the path length of the ϕ to guarantee the image curve of the isogeny is uniform. This means that in their setup protocol, provided a participant is honest, the output j-invariant is uniformly at random in the set of supersingular elliptic curves.

However, there are still some problems with these approaches. The increased challenge space of [BCC^{+}22, DDGZ21] yields a knowledge error of $\frac{2}{3}$ per round,

[5] The supersingular isogeny graph with level d Borel structure, where $d = |\ker \phi|$

which increases the number of repetitions required to achieve a sufficient soundness level. Furthermore, SECUER PoK relies on a relaxed assumption to soundness, namely that the extractor may not obtain the original isogeny ϕ, but an isogeny $\phi' = [\ell_B^{2i}] \circ \phi$ for some $i \leq e_B$.

We forgo these approaches (and their soundness issues) by viewing the isogeny, ϕ, as a walk on the supersingular ℓ_A-isogeny graph and then proving the knowledge of the walk with a generic proof system. Provided the prover can efficiently compute the intermediate j-invariants on the walk, which is done in practice using Vélu's formulae, this provides the same functionality as the proof systems above.

2 Preliminaries

2.1 Notations

A function $f : \mathbb{N} \to \mathbb{R}^+$ is negligible if for every polynomial p there is an N such that for all $n > N$ it holds that $f(n) < \frac{1}{p(n)}$. Given a relation $\mathcal{R}$, we say that $\mathcal{L}(\mathcal{R})$ is the set of all elements x such that there exists a w where $(x, w) \in \mathcal{R}$.

2.2 Isogeny Graphs

This section recalls a few essential properties of supersingular elliptic curves relevant for our work. We refer to [Was08,Sil09] for a more extensive exposition.

Elliptic Curves An elliptic curve is a projective non-singular curve of genus 1. We say a curve is defined over a field K if its coefficients are. The K-rational points, $E(K)$, form a group under an additive operator. Elliptic curves over a field may be uniquely identified (up to isomorphism) by a single field element, called the j-invariant. The j-invariant is efficiently computable given a curve's coefficients.

Isogenies An isogeny is a morphism of elliptic curves preserving both geometric structure (as a rational map) and group structure (as a group homomorphism). The degree of a (separable) isogeny is the size of its kernel as a group homomorphism. We say an isogeny is an ℓ-isogeny if it has degree ℓ, and that two elliptic curves are ℓ-isogenous if there exists an ℓ-isogeny between them. We shall assume all isogenies discussed in this work are separable (but need not necessarily be cyclic).

Supersingular $\boldsymbol{\ell}$-Isogeny Graph We denote the supersingular ℓ-isogeny graph over $\mathbb{F}_{p^2}$ as $G_\ell(p)$, whose vertices are the supersingular elliptic curves over the field (up to isomorphism), with an edge between two vertices if they are ℓ-isogenous. It is a well known fact that for $\ell \neq p$, $G_\ell(p)$ is a Ramanujan graph [Piz90], an optimal expander graph.

Modular Polynomial The modular polynomial $\Phi_\ell(X, Y)$, is a symmetric polynomial of degree $\ell{+}1$ whose roots over $\mathbb{F}_{p^2}$ correspond to every pair of ℓ-isogenous j-invariants of elliptic curves over $\mathbb{F}_{p^2}$. This allows us to efficiently determine if two elliptic curves are ℓ-isogenous over a given field. For $\ell = 2$, we have the modular polynomial

$$\begin{aligned}\Phi_2(X, Y) = X^3 + Y^3 - 162000(X^2 + Y^2) + 1488XY(X + Y) - X^2Y^2 \\ + 8748000000(X + Y) + 40773375XY - 157464000000000. \quad (2)\end{aligned}$$

So, two j-invariants j_1, j_2 are adjacent in $G_\ell(p)$ if and only if $\Phi_\ell(j_1, j_2) = 0 \mod p$.

2.3 Proof Systems

Zero-knowledge succinct Non-interactive Arguments of Knowledge In the (explicitly programmable) random oracle model, a *zero-knowledge non-interactive succinct argument*[6] *of knowledge* (zkSNARK) for a relation $\mathcal{R} = \{(x, w)\}$ is a tuple (P, V) where P, V are probabilistic polynomial time (PPT) algorithms with access to a random oracle ρ which satisfy the following properties:

- Completeness: For every $(x, w) \in \mathcal{R}$, $\lambda \in \mathbb{N}$,
$$\Pr[V^\rho(x, \pi) = 1 \mid \pi \leftarrow P^\rho(x, w)] = 1$$
- Soundness: Given negligible soundness s, for every PPT $\tilde{P}$, $x \notin \mathcal{L}(\mathcal{R})$, and $\lambda \in \mathbb{N}$:
$$\Pr[V^\rho(x, \pi) = 1 \mid \pi \leftarrow \tilde{P}^\rho(x)] \le s(x, \lambda).$$
- Proof of Knowledge: Given negligible knowledge error κ, there exists a PPT extractor E such that, for every x, PPT $\tilde{P}$, $\lambda \in \mathbb{N}$,
$$\Pr[(x, w) \in \mathcal{R} \mid w \leftarrow E^{\tilde{P}}(x, 1^\lambda)] - \Pr[V^\rho(x, \pi) = 1 \mid \pi \leftarrow \tilde{P}^\rho] \le \kappa(x, \lambda).$$
Where the extractor E may program the responses to random oracle queries of $\tilde{P}$, and either get a response of the next query or output π, at which point $\tilde{P}$ goes to the start of its computation with the same randomness and auxiliary input.
- Zero Knowledge: A non-interactive protocol (P, V) is statistical zero-knowledge (with negligible function z) in the explicitly programmable random oracle model (EPRO)[7], if there exists a PPT simulator $\mathcal{S}$, such that for every $(x, w) \in \mathcal{R}$ and unbounded distinguisher D:
$$\Pr[D^{\rho[\mu]}(\pi) = 1 \mid (\pi, \mu) \leftarrow \mathcal{S}^\rho(x)] - \Pr[D^\rho(\pi) = 1 \mid \pi \leftarrow P^\rho(x, w)] \le z(x, \lambda),$$

[6] Typically, a non-interactive random-oracle proof system is a *proof* (NIZKPoK) only if the definition of soundness holds given a computationally unbounded prover, and is otherwise called an *argument*. We may use the terms interchangeably to refer to both.

[7] We include the definition of zero-knowledge in the EPRO model, which is required in the application of the BCS transform—the Fiat-Shamir analogue for IOPs.

where the EPRO, $\rho[\mu]$, outputs $\mu(x)$ if x is in the domain of μ, otherwise it outputs $\rho(x)$. The distributions are taken over the uniformly at random instantiation of ρ and the randomness of P, V.

- Succinctness: A proof system (P, V) for a relation $\mathcal{R}$ is *succinct*, if, for any $(x, w) \in \mathcal{R}$ and corresponding proof $\pi \leftarrow P^{\rho}(x, w)$, π grows polylogarithmically in w. In particular, $|\pi| = \mathsf{poly}(\lambda, |x|, \log(|w|))$.

Interactive Oracle Proofs An *interactive oracle protocol* between two PPT algorithms A and B over k rounds is a protocol where at the ith round, A sends an i-th message m_i to B, who responds with a random access oracle f_i which may be queried in consequent rounds. After k rounds, A either accepts or rejects (see [BCS16] for details).

An *Interactive Oracle Proof* (P, V) for a relation $\mathcal{R}$ with round complexity k and soundess s satifies the following properties:

- Completeness: For every $(x, w) \in \mathcal{R}$, $(P(x, w), V(x))$ is a $k(x)$-round interactive protocol with accepting probability 1.
- Soundness: For every $x \notin \mathcal{L}(\mathcal{R})$ and every $\tilde{P}$, $(\tilde{P}, V(x))$, is a $k(x)$-round interactive oracle protocol with accepting probability at most $s(x)$.

Interactive Oracle Proofs (IOPs), introduced by Ben-Sasson et al [BCS16], are a generalisation of both Interactive Proofs (IPs) and Probabilistically Checkable Proofs (PCPs). One may note that IOPs directly generalise PCPs to multiple rounds. The motivation behind the construction of IOPs is that of efficiency, by minimising redundancy that might be present in a traditional 1 round PCP construction. Analogously to IPs and PCPs, an IOP may also satisfy the properties of zero-knowledge, proof of knowledge, and succinctness, as well as a transformation which performs similarly to the Fiat-Shamir transform [FS87]. Thus, zkSNARKs can be obtained from IOPs. Intuitively, succinct proofs are achievable when the prover sends random access oracles (instantiated via Merkle trees with a CRH function), rather than full length messages.

Theorem 1 (BCS Transform). *There exists a transform T that inputs an IOP (P, V) and outputs a non-interactive argument of knowledge (P^*, V^*) that preserves proof of knowledge and succinctness. Moreover, when the underlying IOP is statistically zero-knowledge, the resulting protocol is statistically zero-knowledge under the EPRO model.*[8]

Proof. See [BCS16, Sec. 6]

In this work, we consider IOPs that satisfy all of these properties and are also *transparent*. That is, secure in the absence of the common reference string (CRS) model, in which protocols require trusted setup.

[8] In particular, the extractor in the transformation T is straight-line, and does not apply the forking lemma.

2.4 Rank-1 Constraint Systems

We recall the definition of rank-1 constraint systems (R1CS), which some zk-SNARKs (e.g., Aurora) take as an input. R1CS is parameterized by $n, m \in \mathbb{N}$ and a prime power q, and consists of instance-witness pairs $((A, B, C, v), w)$ where $A, B, C \in \mathbb{F}_q^{m\times(n+1)}$ and v,w are vectors over $\mathbb{F}_q$ such that

$$Az \circ Bz = Cz$$

for $z := (1, v, w) \in \mathbb{F}_q^{n+1}$, where $\circ$ denotes coordinate-wise (Hadamard) product. Conceptually, A, B, C encode constraints on variables v, w; where v contains (public) auxiliary input, and w contains both secret input and intermediate variables in a computation.

R1CS typically encodes arithmetic circuit satisfiability. However, we work with modular polynomials and show how to an isogeny path relation directly into a R1CS together with some optimisations in Sec. 3.4.

2.5 MPC-in-the-Head

The MPC-in-the-Head (MPCitH) paradigm was introduced in the seminal work of Ishai et al. [IKOS07] Suppose the prover wishes to convince a verifier of an NP relation $\mathcal{R}$ in zero-knowledge, where x is the instance and w is the witness. The prover simulates a semi-honest MPC protocol with n parties locally (in its head) and commits to the transcript. The verifier asks the prover to decommit a subset of the transcript and check whether the messages are consistent and that the reconstructed output is 1, meaning that the relation $\mathcal{R}$ holds. If there are no failures during the verification, the verifier accepts the proof. Intuitively, completeness holds trivially, (statistical) zero-knowledge holds if the decommitted transcript is not enough to reveal the full transcript (e.g., revealing $n - 1$ transcripts reveals nothing about the full transcript when using additive secret sharing). Regarding soundness, the prover may cheat if the faulty transcript is not challenged by the verifier. Nevertheless, it is possible to boost the soundness by repeating the protocol many times. Using the Fiat-Shamir transform [FS87] it is possible to convert an interactive protocol to a non-interactive one.

Limbo Limbo [dOT21] is the state-of-the-art non-interactive zero-knowledge proof of knowledge for arithmetic circuit satisfiability protocol based on the MPCitH paradigm. Despite not satisfying the asymptotic definition of succinctness, Limbo has proven to have good concrete efficiency for small to medium sized circuits (i.e. circuits with less than 500000 multiplication gates). Thus we include it in our consideration. For the detailed description, we refer the reader to the paper.

2.6 Reed-Solomon IOPs

The other line of protocols [BCR+19, AHIV17, BFH+20] we consider in this work is called Reed-Solomon IOPs. In contrast to the MPCitH-based approach above, these protocol achieve the property of succinctness.

At a high level, in RS-IOPs, the witness w corresponds to the input plus all the intermediate variables in the computation. The prover transforms the witness w into various vectors, depending on the proof system, which are then encoded with a RS code. The verifier engages in various sub-protocols with the prover to check conditions on the RS encoded values to convince itself that the encoded values form valid RS codewords and satisfies the constraints given in the relation.

Reed-Solomon Codes Given an ordered subset $L = \{\ell_1, ..., \ell_k\}$ of a field $\mathbb{F}_q$ and $\alpha \in (0, 1]$, we denote $RS[L, \alpha] \subseteq \mathbb{F}_q^k$ to be the set of evaluations over L of all polynomials of degree less than αk. That is, a codeword c is in $RS[L, \alpha]$ if and only if there exists a polynomial p of degree less than αk such that the $c = (p(\ell_1), ..., p(\ell_k))$.

Aurora Aurora is a transparent zkSNARK for the R1CS relation secure in the EPRO. At a high level, Aurora's underlying IOP reduces to proving the following two subproblems:

- Rowcheck: Given vectors $a, b, c \in \mathbb{F}_q^m$, test whether $a \circ b = c$
- Lincheck: Given vectors $x \in \mathbb{F}_q^m$, $y \in \mathbb{F}_q^{n+1}$, and matrix $M \in \mathbb{F}_q^{m\times(n+1)}$; test whether $x = My$.

Given IOPs for these problems, one may construct an IOP for R1CS. Given an R1CS instance $((q, n, m, A, B, C, v), w)$, the prover sends four oracles to the verifier: the satisfying assignment for z, $y_A := Az$, $y_B := Bz$, and $y_C := Cz$. The prover then engages in parallel execution of the following:

- An IOP for Rowcheck to verify that $y_A \circ y_B = y_z$.
- An IOP for Lincheck to verify that $y_A = Az$, $y_B = Bz$, and $y_C = Cz$.

Finally, the verifier checks that z is consistent with the auxiliary input v.

However, such a protocol would be neither succinct, nor zero-knowledge. In order for the protocol to achieve sublinear communication complexity, the subprotocols for Lincheck and Rowcheck both utilise Reed-Solomon encoded variants. In this case, foregoing zero-knowledge, the subroutines for Lincheck and Rowcheck encode the vectors y_A, y_B, y_C as the coefficients of a unique polynomial that matches them over some $H_1 \subset \mathbb{F}_q$ where $|H_1| = m$, and likewise for z, as the coefficients of a polynomial that matches z over some $H_2 \subseteq \mathbb{F}_q$ where $|H_2| = n + 1$. In addition, some extra work is done to check the degree of the polynomials is consistent with the input via a *low-degree* test. Aurora utilises the FRI protocol [BBHR18] to achieve this efficiently.

Zero knowledge is achieved by encoding a vectors Az, Bz, Cz not as unique polynomial of degree $|H_1|-1$ matching the entries of Az, Bz, Cz on H_1, but as a random polynomial of degree $|H_1|+m$ conditioned on matching Az, Bz, Cz on H_1 (the same process applies to z with domain H_2). The polynomial is represented as evaluations over a domain L disjoint from H_1 and H_2 such that m queries cannot leak any information about v. In order to guarantee these subsets are disjoint, over a prime field, the subsets H_1, H_2 are chosen to be multiplicative subgroups of the field (of order a power of two such that $H_1 \subseteq H_2$ or $H_2 \subseteq H_1$), and the evaluation domain L is a multiplicative coset of a subgroup of $H_1 \cup H_2$.

Ligero Ligero [AHIV17] is another transparent zkSNARK based on a RS-IOP for boolean or arithmetic circuit satisfiability (technically, an IPCP, as it only comprises of a single round). Given an arithmetic circuit C of N gates, a Ligero prover represents the satisfying assignment of the s ($\approx N$) wires of C into a slightly redundant matrix representation of size $O(\sqrt{s}) \times O(\sqrt{s})$, and encodes each row of this matrix using an (interleaved) RS code. The verifier challenges the prover to reveal linear combinations of the entries of the codeword matrix, which is checked against λ randomly selected columns of the matrix which are consequently revealed by the prover.

Aside from the underlying proof relation, the key distinction between Aurora and Ligero is informed by two design decisions: Ligero encodes its oracles with $O(\sqrt{N})$ RS codewords of length $O(\sqrt{N})$, rather than by a single RS codeword of length $O(N)$. In addition, it uses a direct (single-round) low-degree test rather than the FRI IOP.

3 Construction

3.1 Hardness assumptions and relations

Recent attacks have rendered the SIDH assumption broken [CD22] [Rob22]. The key insight is that the Castryck-Decru and Robert attacks require the image of the torsion points P_1, Q_1, however, the following, more general isogeny path-finding problem below, historically used to cryptanalyse SIDH, remains unaffected.

*Problem 1 (*IsoPath*).* Given supersingular elliptic curves E_0, E_1 defined over $\mathbb{F}_{p^2}$, find an isogeny $\phi : E_0 \to E_1$ such that $\deg \phi = \ell^k$ for a fixed prime ℓ and $k \in \mathbb{Z}$.

We define the following relation based on the hardness of IsoPath:

$$\mathcal{R}_{\ell^k\text{-IsoPath}} = \{((E_0, E_1), \phi) \ : \ \phi : E_0 \to E_1 \text{ is an isogeny, } \deg \phi = \ell^k, k \in \mathbb{Z}\}$$

The isogeny witness ϕ is typically represented by fixing a basis of the ℓ^k-torsion group, and giving a kernel generator, a point on E_0 of order ℓ^k. Instead, we choose to represent our witness isogeny ϕ in the relation above by using the modular polynomial. Recall, two elliptic curves E, E' are ℓ-isogenous if and only if $\Phi_\ell(j(E), j(E')) = 0$. Then an isogeny $\phi : E_0 \to E_1$ of degree ℓ^k can equivalently be represented as a sequence of intermediate j-invariants $j_1, j_2, ..., j_{k-1}$ such that

$$\begin{aligned} \Phi_\ell(j(E_0), j_1) &= 0 \\ \Phi_\ell(j_i, j_{i+1}) &= 0 \quad \text{for all } i \in \{1, ..., k-2\} \\ \Phi_\ell(j_{k-1}, j(E_1)) &= 0 \end{aligned}$$

Hence, more precisely, the relation we prove is as follows:

$$\mathcal{R}_{\ell^k\text{-ModPoly}} = \left\{ \left((E_0, E_1), (j_i)_{i \in \{1,\ldots,k-1\}}\right) \ : \ \begin{array}{l} \Phi_\ell(j(E_0), j_1) = 0, \Phi_\ell(j_{k-1}, j(E_1)) = 0 \\ \Phi_\ell(j_i, j_{i+1}) = 0 \quad \forall i \in \{1, ..., k-2\} \end{array} \right\}$$

When generating isogeny path instances, we want the length k to be small enough to be efficient, but large enough to prevent meet-in-the-middle and collision search claw-finding type attacks [Gal99,vW99,ACC$^+$19], whose classical and quantum heuristic run times are $\tilde{O}(\ell^{k/2})$ and $\tilde{O}(\ell^{k/3})$ respectively. One might therefore take $k \approx 2\lambda$ as reasonable security trade-off.

Remark 1. Note that in this case, the isogeny may not necessarily be cyclic. In fact, the isogeny walk taken could indeed contain backtracking. In the applications we discuss in this work, this is not a problem, since an honest prover would honestly generate a non-backtracking isogeny of degree k, which would hence be cyclic. If one wishes to guarantee non-backtracking walks, this problem can be resolved by adding the requirement that $j_{i-1} \neq j_{i+1}$ for all i in $\{1, ..., k-1\}$. We explain how to prove this with a cheap overhead in App. A.

3.2 High-Level Overview

The reader might wonder, what in particular does our isogeny representation achieve? What makes this relation so amenable to generic proof systems is its low-depth, highly *regular* decision circuit. That is, an arithmetic circuit C where $C(x, w) = 1$ if and only if $(x, w) \in \mathcal{R}_{\ell^k\text{-ModPoly}}$. In this case, C may simply be a sequence of parallel evaluations of the modular polynomial on each pair of adjacent j-invariants. This allows us encode the relation in a highly compact (but equivalent) intermediate representation, to be fed into the proof system.

The general roadmap to utilising the generic proof systems is as follows:

1. Encode the relation $\mathcal{R}_{\ell^k\text{-ModPoly}}$ and pair (x, w) into an equivalent R1CS, denoted by $\mathcal{R}'_{\ell^k\text{-ModPoly}}$ and (x', w') respectively.
2. Use a generic zkSNARK for R1CS (resp. arithmetic circuits) to argue the knowledge of a witness w' such that $(x', w') \in \mathcal{R}'_{\ell^k\text{-ModPoly}}$.
3. The prover's knowledge of w' will imply the knowledge of w such that $(x, w) \in \mathcal{R}_{\ell^k\text{-ModPoly}}$.

Since we have to perform field arithmetic over a quadratic extension field we can either work over the base field (where each $\mathbb{F}_{p^2}$-multiplication will dictate a series of underlying $\mathbb{F}_p$ multiplications), or adapt the proof system implementation to be suitable for quadratic extensions. The security of the proof systems in question are independent of field choice, but the efficiency of Reed Solom based protocols is subject to a requirement. Namely, being capable of performing efficient FFT and IFFT operations. Broadly speaking, working over a field K, we require that $K^\times$ contains a subgroup of order 2^m for an integer m such that $c \leq 2^m$, where $c = \max\{m, n\}$ for n variables and m constraints in a given R1CS. When working with isogenies, we typically choose primes of the form

$p_1 = 2^a 3^b f - 1$, or $p_2 = 2^a 3^b f + 1$. It is clear that $\mathbb{F}_{p_2}$ would satisfy the condition above, provided $m \leq a$, but $\mathbb{F}_{p_1}$ would not, since $|\mathbb{F}_{p_1}^{\times}| = p_1 - 1 = 2(2^a 3^b f - 1)$. The first solution is to simply instantiate the proof system only over the base field with p_2 primes, however this admits several problems. Firstly, $\mathbb{F}_{{p_1}^2}$ operations are slightly more efficient. Since -1 is a non quadratic residue, $\mathbb{F}_{{p_1}^2} \cong \mathbb{F}(i)$ which allows for more efficient multiplication, inversion and squarings. Secondly, we want our protocol to be compatible with common choices of parameters, which typically use p_1 primes for efficiency reasons. Thus, we instantiate the proof system over the extension field, whose multiplicative order is $p^2 - 1 = (p-1)(p+1)$. This satisfies either choice of prime.

3.3 From Isogeny Relation to R1CS Instance

In order to apply our proof systems, we transform the modular polynomial relation into an R1CS with n variables and m constraints. Concretely, we consider an R1CS consisting of the statement $A, B, C \in \mathbb{F}_{p^2}^{m \times (n+1)}$ and a witness $z \in \mathbb{F}_{p^2}^{n+1}$ such that

$$Az \circ Bz = Cz.$$

In this formulation, A, B, C are public matrices which correspond to an instantiation of the language dependent on p, ℓ, k. The vector z consists of 1, the auxiliary input: j-invariants of the starting and ending curve, and the secret input: the j-invariant sequence (as well as intermediate variables dependent on the inputs). Each row of A, B, C will encode a linear constraint on the variables. One of these rows must encode the isogeny modular polynomial $\Phi_\ell(j_i, j_{i+1}) = 0$, which shows that two adjacent j-invariants are isogenous. For representation compactness, we arrange the modular polynomial in the following form:

$$\begin{aligned} -1488XY(X + Y - 1488^{-1}XY) = X^3 + Y^3 - 162000(X^2 + Y^2) + \\ 8748000000(X + Y) + 40773375XY - 157464000000000 \end{aligned} \quad (3)$$

3.4 Optimization for R1CS over $\mathbb{F}_{p^2}$

We then encode matrices A, B, C such that a row evaluates the equation above and performs intermediate variable consistency checks. Note that we can do far better than the naive approach, where each row of the matrices correspond to a single multiplication or addition of variables in z, and the entries of z contain every intermediate variable obtained. In loose terms, in R1CS, each row can encode: *linear expression* × *linear expression* = *linear expression.*

Suppose the isogeny path in question is of length k. If $k = 1$, $\ell = 2$ then by Eq. (2), we obtain:

$$z = \begin{pmatrix} 1 & j_0 & j_1 & j_0^2 & j_1^2 & j_0^3 & j_1^3 & j_0 j_1 \end{pmatrix}^T$$

with the matrices:

$$A = \begin{bmatrix} 0&1&0&0&0&0&0&0 \\ 0&0&1&0&0&0&0&0 \\ 0&0&0&1&0&0&0&0 \\ 0&0&0&0&1&0&0&0 \\ 0&1&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&-1 \end{bmatrix}, \quad B = \begin{bmatrix} 0&1&0&0&0&0&0&0 \\ 0&0&1&0&0&0&0&0 \\ 0&1&0&0&0&0&0&0 \\ 0&0&1&0&0&0&0&0 \\ 0&0&1&0&0&0&0&0 \\ 0&c_4&c_4&0&0&0&0&-1 \end{bmatrix}, \quad C = \begin{bmatrix} 0&0&0&1&0&0&0&0 \\ 0&0&0&0&1&0&0&0 \\ 0&0&0&0&0&1&0&0 \\ 0&0&0&0&0&0&1&0 \\ 0&0&0&0&0&0&0&1 \\ c_0&c_1&c_1&c_2&c_2&1&1&c_3 \end{bmatrix}$$

where

$$\begin{array}{lll} c_0 = -157464000000000 & c_1 = 8748000000 & c_2 = -162000 \\ c_3 = 40773375 & c_4 = 1488, & \end{array}$$

where the c_i's are derived from Eq. (3). The first 5 rows provide consistency checks on each variable, including square, cube, and multiplication. The last row checks the evaluation of the polynomial Eq. (3). Now we can to extend this to a path of length $k > 1$, for each j-invariant j_i, we will introduce an additional 4 variables (including input): j_i, j_i^2, j_i^3, $j_{i-1}j_i$. We note that the squarings and cubings for each j-invariant need only be checked once. Hence, we obtain $n := 4k + 3$ variables.

For each j-invariant in the sequence (including j_0) there will be 2 constraints for squaring and cubing consistency checks. For each adjacent pair j_{i-1}, j_i, there will be 2 constraints: one checking consistency of the variable $j_{i-1}j_i$, and one the evaluation of the modular polynomial. This gives us $m := 4k + 2$ constraints.

3.5 Optimization for Lifting to $\mathbb{F}_p \times \mathbb{F}_p$

This subsection presents several techniques to reduce the overhead to lift arithmetic over a quadratic field to a vector space of the prime field. We consider a quadratic field $\mathbb{F}_{p^2} \cong \mathbb{F}_p[\alpha]$ where $\alpha^2 = d$ for some non-square $d \in \mathbb{F}_p$.

The motivation is that, generally, the j-invariant of an elliptic curve is taken over $\mathbb{F}_{p^2}$ while some proof systems only support arithmetic over a prime field. Indeed, arithmetic compuations over $\mathbb{F}_p[\alpha]$ can be viewed as arithmetic computations over an $\mathbb{F}_p$-vector space natively. That is, for $x_1, x_2, y_1, y_2 \in \mathbb{F}_p$ to represent $x_1 + x_2\alpha \in \mathbb{F}_p[\alpha]$, by mapping $x_1 + x_2\alpha$ to (x_1, x_2) the addition is $(x_1 + y_1, x_2 + y_2)$ and the multiplication is $(x_1y_1 + x_2y_2d, x_1y_2 + x_2y_1)$. Naively, this results in 4 (variable) $\mathbb{F}_p$-multiplications for one (variable) $\mathbb{F}_{p^2}$-multiplication (i.e. $x_1x_2, y_1y_2, x_1y_2, x_2y_1$). In fact, with a few well-known tricks, this can be done more efficiently:

Arithmetic. We start with multiplications.

- $u_1 = x_1y_1$
- $u_2 = y_2y_2$
- $u_3 = (x_1 + x_2)(y_1 + y_2)$, then
- $x_1y_1 + x_2y_2d = u_1 + u_2d$
- $(x_1y_2 + x_2y_1) = u_3 - u_1 - u_2$

By using the trick, there are only 3 (variable) $\mathbb{F}_p$-multiplications now. The saving depends on the proof system to be used. In many proof systems, it is much more expensive to verifiy a (variable) multiplication relation than a (variable) linear relation.

Let $x + y\alpha \in \mathbb{F}_p[\alpha]$, there is a trick for variable squaring:

- $u_1 = xy$
- $u_2 = (x + y)(x + yd)$, then
- $(x^2 + y^2i^2) = u_2 - (d + 1)u_1$
- $2xy = 2u_1$

Application to R1CS Matrices. Now we can apply the abovementioned techniques to our R1CS matrices. Recall that in Sec. 3.4, we have a witness vector z over $\mathbb{F}_p \times \mathbb{F}_{p^2}^7$. To lift it into $\mathbb{F}_p$, we firstly naturally embed it into $\mathbb{F}_p \times \mathbb{F}_p^{14}$. We explain how to build a submatrices and introduce intermediate variables for each constraint as follows. As an abuse of notation, given an element $x := a + b\alpha \in \mathbb{F}_p[\alpha]$, we refer to a as $\mathsf{Re}(x)$ and b as $\mathsf{Im}(x)$ respectively.

Squaring. For the squaring relation, it is fairly simple. Take the subvector $(1, \mathsf{Re}(x), \mathsf{Im}(x), \mathsf{Re}(x^2), \mathsf{Im}(x^2))$ for instance, the corresponding submatrices for this constraint are respectively

$$\begin{bmatrix} 0 & 2 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & d & 0 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 2^{-1}(d+1) \end{bmatrix},$$

which represents $2\mathsf{Re}(x)\mathsf{Im}(x) = \mathsf{Im}(x^2)$ and $(\mathsf{Re}(x) + \mathsf{Im}(x))(\mathsf{Re}(x) + d\mathsf{Im}(x)) = \mathsf{Re}(x^2) + 2^{-1}(d+1)\mathsf{Im}(x^2)$, resp.

Multiplication. For the multiplication relation, we need an additional variable u over $\mathbb{F}_p$. We take the subvector $(1, \mathsf{Re}(x), \mathsf{Im}(x), \mathsf{Re}(y), \mathsf{Im}(y), u, \mathsf{Re}(xy), \mathsf{Im}(xy))$ for instance. The corresponding submatrices for this constraint are respectively

$$\begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -d & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1-d & 1 & 1 \end{bmatrix},$$

which represents $\mathsf{Im}(x)\mathsf{Im}(y) = u$, $\mathsf{Re}(x)\mathsf{Re}(y) = \mathsf{Re}(xy) - ud$, and $(\mathsf{Re}(x) + \mathsf{Im}(x))(\mathsf{Re}(y) + \mathsf{Im}(y)) = \mathsf{Im}(xy) + \mathsf{Re}(x)\mathsf{Re}(y) + u$, respectively.

Constraint Eq. (3). We can apply our multiplication technique above to the constraint Eq. (3). Recall that the final constraint from the modular polynomial is $(-xy)(c_4x + c_4y - xy) = x^3 + y^3 + c_2(x^2 + y^2) + c_1(x + y) + c_3xy + c_0$. The insight is every coefficient c_i is over $\mathbb{F}_p$ so $\mathsf{Re}(\cdot)$ has the linear proposition $\mathsf{Re}(c_4x + c_4y - xy) = c_4\mathsf{Re}(x) + c_4\mathsf{Re}(y) - \mathsf{Re}(xy)$ and so does the imaginary part $\mathsf{Im}(\cdot)$. Therefore, we can use three constraints for the real part and the imaginary part of $x^3 + y^3 + c_2(x^2 + y^2) + c_1(x + y) + c_3xy + c_0$ in terms of $\mathsf{Re}(X), \mathsf{Im}(X), \mathsf{Re}(Y), \mathsf{Im}(Y)$ where $X = -xy$ and $Y = c_4x + c_4y - xy$ as the method described above.

Concretely, for a subvector

$$z' = (1\ \mathsf{Re}(x)\ \mathsf{Im}(x)\ \mathsf{Re}(y)\ \mathsf{Im}(y)\ \mathsf{Re}(x^2)\ \mathsf{Im}(x^2)\ \mathsf{Re}(y^2)\ \mathsf{Im}(y^2)$$
$$\mathsf{Re}(x^3)\ \mathsf{Im}(x^3)\ \mathsf{Re}(y^3)\ \mathsf{Im}(y^3)\ \mathsf{Re}(xy)\ \mathsf{Im}(xy)\ u)$$

the corresponding submatrices for this constraint are respectively

$$\begin{bmatrix} 0&0&0&0&0&0&0&0&0&0&0&0&0&0&-1&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0&-1&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0&-1&-1&0 \end{bmatrix}, \begin{bmatrix} 0&0&c_4&0&c_4&0&0&0&0&0&0&0&0&0&-1&0 \\ 0&c_4&0&c_4&0&0&0&0&0&0&0&0&0&-1&0&0 \\ 0&c_4&c_4&c_4&c_4&0&0&0&0&0&0&0&0&-1&-1&0 \end{bmatrix},$$

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ c_0 & c_1 & 0 & c_1 & 0 & c_2 & 0 & c_2 & 0 & 1 & 0 & 1 & 0 & c_3 & 0 & -d \\ c_0 & c_1 & c_1 & c_1 & c_1 & c_2 & c_2 & c_2 & c_2 & 1 & 1 & 1 & 1 & c_3 & c_3 & (1-d) \end{bmatrix},$$

which respectively represents

$$\begin{aligned} \mathsf{Im}(X)\mathsf{Im}(Y) &= u \\ \mathsf{Re}(X)\mathsf{Re}(Y) &= \mathsf{Re}(Z) - ud \\ (\mathsf{Re}(X) + \mathsf{Im}(X))(\mathsf{Re}(Y) + \mathsf{Im}(Y)) &= \mathsf{Im}(Z) + \mathsf{Re}(Z) + (1-d)u, \end{aligned}$$

where

$$\begin{aligned} X &= -xy \\ Y &= c_4x + c_4y - xy \\ Z &= x^3 + y^3 + c_2(x^2 + y^2) + c_1(x + y) + c_3xy + c_0. \end{aligned}$$

In summary, for any isogeny path over any quadratic field $\mathbb{F}_{p^2}$ of length k, we can transform it into an R1CS relation with $11k+4$ variables and $11k+3$ constraints over $\mathbb{F}_p$.

3.6 Parameter choice

In order to offer a wider degree of flexibility, we apply our R1CS relation over both $\mathbb{F}_p$ and $\mathbb{F}_{p^2}$ arithmetic, which allows for the support of:

- isogeny-based protocols (working over $\mathbb{F}_{p^2}$) with primes of the form $p = 2^a3^bf + 1$ with proof system operating over $\mathbb{F}_p$,
- and isogeny-based protocols with primes of the form $p = 2^a3^bf \pm 1$ operating over $\mathbb{F}_{p^2}$.

Once an isogeny path has been obtained, it is straightforward to obtain either R1CS instance given the methods described in Sec. 3.4 and Sec. 3.5. We leave the manner in which the isogeny paths are computed open to a more detailed implementation. One such approach would be to use optimized SIDH implementations [CLN16, ACC+17], with some modifications needed to support p_2 primes. Note that since $p_2 \equiv 1 \mod 4$, the curves of j-invariant $0, 1728$ are not supersingular. In this setting, one can find a starting curve by using a root of the Hilbert class polynomial mod p [Brö08, Sec 3.2]. The public parameters p, ℓ, k are sufficient for a verifier to efficiently construct the R1CS matrices A, B, C offline, which minimises the communication and storage cost.

In evaluating performance for comparison with [BCC+22], we have included the standard SIKE parameters, but also include primes of comparable parameters of the p_2 form in order to compare performance of over different base fields, which should offer equivalent security at the cost of slightly reduced performance of isogeny path computation. These primes are the smallest primes $p_2 = 2^a3^bf+1$ such that for a corresponding SIKE prime $p_1 = 2^{a'}3^{b'}f' - 1$ we have that $a \geq a'$, $b \geq b'$ and $f' \geq f$. Due to the flexibility of the underlying proof systems, the protocol can operate over arbitrary choices of k, and primes of this form. We have fixed the path length k to the corresponding lengths of the Secuer PoK 2-isogeny path, which is of sufficient length to guarantee a uniformly distributed end point, assuming a random walk. In fact, this is a conservative choice. It is conjectured that non-backtracking walks can converge to the stationary distribution in shorter walks than compared to [BCC+22]. See [Ste22, Conjecture 4.3]. We stress that in many applications, uniform mixing is not necessary. In order to guarantee minimal security, the path's length must be approximately 2λ. For the parameters and results of our proof for minimally secure path lengths, see Sec. 4.2.

Remark 2. The choice of benchmarking this protocol with parameters obtained from the now defunct SIKE may seem somewhat arbitrary. We do so to compare our results to [BCC+22], whose implementation is limited to SIKE primes. A pragmatic course of action might be to determine concrete parameters that are practical and secure in the setting of isogeny commitments and hashing.

			R1CS Param.		
p	k	Variant	n	m	Security Level
`p434` $2^{216}3^{137}-1$	705	$\mathbb{F}_{p^2}$	2823	2822	$\lambda = 128$
`p441+` $2^{218}3^{138}37+1$		$\mathbb{F}_p$	7759	7758	
`p503` $2^{250}3^{159}-1$	774	$\mathbb{F}_{p^2}$	3099	3098	$\lambda = 128$
`p509+` $2^{252}3^{159}31+1$		$\mathbb{F}_p$	8518	8517	
`p610` $2^{305}3^{192}-1$	1010	$\mathbb{F}_{p^2}$	4043	4044	$\lambda = 192$
`p619+` $2^{307}3^{192}119+1$		$\mathbb{F}_p$	11114	11113	
`p751` $2^{372}3^{239}-1$	1280	$\mathbb{F}_{p^2}$	5123	5122	$\lambda = 256$
`p761+` $2^{372}3^{239}701+1$		$\mathbb{F}_p$	14084	14083	

Table 1. Our parameter sets for the evaluation of isogeny PoK in R1CS representation.

4 Implementation and Evaluation

In evaluating the performance of our isogeny proof of knowledge, we considered protocols which support finite fields of prime characteristic[9], that are statistical zero-knowledge, plausibly post-quantum and transparent (see Tab. 2).

Virgo and Orion [ZXZS19, XZS22] do satisfy these properties. However, we excluded them from our testing as their implementation does not easily support generic fields, but we hope to include them in future testing. Theoretically, Virgo performs well for low-depth, uniform circuits such as our own.

[9] Subject to FFT performance conditions.

The state-of-the-art is given by Ligero++; a protocol that combines aspects of Virgo and Ligero, trading-off marginally higher verification times for faster prover times than Aurora, with comparable proof sizes. However, it does not have any open source implementations. Brakedown, Shockwave [GLS$^+$21]; and the recent LaBRADOR [BS22] are candidates of interest. However, they do not yet offer zero-knowledge. There are no clear obstructions to them achieving zero-knowledge, and provide promising results, so are worth considering in future lines of work.

	Prover time	Verifier time	Proof size
Limbo [dOT21]	$O(N)$	$O(N)$	$O(N)$
Ligero [AHIV17]	$O(N \log N)$	$O(N)$	$O(\sqrt{N})$
Aurora [BCR$^+$19]	$O(N \log N)$	$O(N)$	$O(\log^2 N)$
Virgo [ZXZS19]	$O(N + n \log n)$	$O(D \log N + \log^2 n)$	$O(D \log N + \log^2 n)$
Ligero++ [BFH$^+$20]	$O(N \log N)$	$O(N)$	$O(\log^2 N)$
Orion [XZS22]	$O(N)$	$O(\log^2 N)$	$O(\log^2 N)$

Table 2. Asymptotic cost various transparent, post-quantum, zero-knowledge generic proof systems, applied to an arithmetic circuit of N gates, n inputs, and depth D over a fixed field.

Implementation As a proof of concept, we evaluate the performance of our isogeny proof of knowledge via:

- **Aurora** and **Ligero** through a fork of `libiop`[10], modified to support larger prime fields and quadratic field extensions. Ligero's original implementation is closed source, but an adaptation is included in `libiop`. While originally designed for arithmetic circuit satisifiability, `libiop`'s implementation supports R1CS instead, at claimed *no extra cost.*
- **Limbo**, through an implementation obtained via private correspondence (the publicly available implementation is only available for binary fields). Limbo is interfaced with our R1CS instances directly, with an arithmetic circuit that evaluates $Az \circ Bz - Cz$ and then checking that the resulting vector equals to zero.

Aurora and Ligero are directly tested with R1CS instances of size given in Tab. 1. We separate the results for the standard SIKE parameters for direct comparison with Secuer PoK, and include a second table of results (Tab. 4) for the smooth primes which operate over $\mathbb{F}_p$. Limbo is directly interfaced to prove the given R1CS instance in a manner described in Sec. 2.5.

[10] Original source code available at `https://github.com/scipr-lab/libiop`. Our fork can be found at `https://github.com/levanin/libiop-other-primes`.

4.1 Comparison to Secuer PoK

To make a comparison, we include the results from the **Secuer PoK** [BCC$^+$22], through the reference implementation[11]. The Secuer PoK is a direct proof of knowledge for a relaxed notion of the relation $\mathcal{R}_{2^k\text{-IsoPath}}$, so provides comparison as a tailored protocol to our results from applying generic proof systems.

Remark 3. In the previous version, our implementations are inconsistent with [BCC$^+$22] regarding the walk length. The Secuer PoK reports results for walks of sufficient length in order to guarantee uniform mixing in the supersingular isogeny graph. For consistency, we now evaluate our results based on the same parameters. This is a desired feature in the setup ceremony protocol introduced in their work. However, this is not a strict requirement in isogeny-based protocols. We include an additional set of results in Sec. 4.2 which include results for walk lengths which are minimally secure for the respective security levels, which may be of interest in wider applications.

		Our Work			
Parameter		Aurora	Ligero	Limbo	Secuer PoK
`p434`	P	4,204ms	1,479ms	1,073ms	12,369ms
	V	378ms	1,899ms	874ms	1,399ms
	S	277kB	3,281kB	8,133kB	191kB
`p503`	P	4,944ms	1,722ms	1,379ms	19,296ms
	V	440ms	2,171ms	1,146ms	2,173ms
	S	313kB	3,778kB	10,335kB	216kB
`p610`	P	6,457ms	3,331ms	3,156ms	60,915ms
	V	888ms	3,102ms	2,616ms	6,646ms
	S	570kB	4,568kB	24,427kB	404kB
`p751`	P	12,555ms	5,243ms	7,702ms	141,043ms
	V	1651ms	13,509ms	6,587ms	15,931ms
	S	688kB	11,302kB	50,670kB	663kB

Table 3. Table of results comparing several generic proof systems operating over $\mathbb{F}_{p^2}$ for the R1CS instantiation of $\mathcal{R}_{2^k\text{-MP}}$, and the isogeny Seceur PoK in [BCC$^+$22]. Security level and walk length is set according to Tab. 1 and P, V, S correspond to proof time, verification time, and proof size respectively. Results displayed are for single-threaded performance.

Results The experiments are run on a Intel® Core™ i9-9900 CPU @ 3.10GHz. The benchmarks include only single-threaded results as the `libiop` package does

[11] Source code available at `https://github.com/trusted-isogenies/SECUER-pok`

Parameter		Aurora	Ligero	Limbo
`p441+`	P	2,313ms	879ms	1,037ms
	V	158ms	1017ms	835ms
	S	152kB	2,803kB	11,217kB
`p509+`	P	5,999ms	1301ms	1,304ms
	V	455ms	1370ms	1,066ms
	S	214kB	3,402kB	14,205kB
`p619+`	P	9,424ms	2,822ms	2,962ms
	V	895ms	2,030ms	2,451ms
	S	409kB	4,149kB	33,746kB
`p761+`	P	12,555ms	2,873ms	7,062ms
	V	1,651ms	4,464ms	5,823ms
	S	687kB	7,212kB	70,018kB

Table 4. Table of results comparing generic proof systems operating over $\mathbb{F}_p$ for the projected R1CS instantiation of $R_{2^k\text{-MP}}$ operating over fields with characteristic of the form $2^a 3^b f + 1$. Security level and walk lengths set according to Tab. 1. Results displayed are for single threaded performance.

not properly implement multi-threading and did not provide accurate results. Nevertheless, Aurora and Ligero should reflect similar optimizations to that of [BCC+22] from a well supported multi-threaded implementation, as the protocols are well suited to parallelisation. In particular, the protocols run parallel compositions of the proof in order to achieve necessary soundness level.

We see that Aurora, the best overall performer, provides a 3-12 times improvement to proof and verification times compared to SECUER PoK, with 0-30% increase in proof length. If we consider smooth primes which allow for operation over $\mathbb{F}_p$, Aurora allows for similar improvements to proof and verification times but with smaller proofs than SECUER PoK when compared with parameters of similar bit length. Limbo, as expected, performs well for smaller parameters at the cost of much longer proof lengths. Conversely, Ligero is better suited to larger parameters than Limbo but still suffers from long proofs. These results should serve as evidence to support the choice of Aurora as a platform for this application.

4.2 Identification Scheme for Moderate Length Walks

Our proof system may also serve as an identification scheme to validate a public key (E, E'), where the prover can use our zkSNARK construction to demonstrate their knowledge of a walk from E to E', of sufficient length to resist the most efficient generic algorithm for recovering the secret isogeny (i.e. the claw finding algorithm). In this section, we demonstrate the effectiveness of our proof system in this regard.

We show in Tab. 5 the R1CS parameter set (m, n) over $\mathbb{F}_p$ and $\mathbb{F}_{p^2}$ and the isogeny walk length k with respect to the security parameter λ. A concrete result is given in Tabs. 6 and 7 regarding the prover time, the verifier time and the proof size for different forms of the primes.

			R1CS Param.		
p	k	Variant	n	m	Security Level
`p434` $2^{216}3^{137}-1$	216	$\mathbb{F}_{p^2}$	867	866	$\lambda = 128$
`p441+` $2^{218}3^{138}37+1$		$\mathbb{F}_p$	2380	2379	
`p503` $2^{250}3^{159}-1$	250	$\mathbb{F}_{p^2}$	1003	1002	$\lambda = 128$
`p509+` $2^{252}3^{159}31+1$		$\mathbb{F}_p$	2754	2753	
`p610` $2^{305}3^{192}-1$	305	$\mathbb{F}_{p^2}$	1223	1222	$\lambda = 192$
`p619+` $2^{307}3^{192}119+1$		$\mathbb{F}_p$	3359	3358	
`p751` $2^{372}3^{239}-1$	372	$\mathbb{F}_{p^2}$	1491	1490	$\lambda = 256$
`p761+` $2^{372}3^{239}701+1$		$\mathbb{F}_p$	4096	4095	

Table 5. Our R1CS parameter set (m, n) over $\mathbb{F}_p$ and $\mathbb{F}_{p^2}$ and the isogeny walk length k with respect to the security parameter λ and the prime p.

		Our Work		
Parameter		Aurora	Ligero	Limbo
p434	P	934ms	587ms	354ms
	V	99ms	847ms	273ms
	S	194kB	1,849kB	2,598kB
p503	P	1,138ms	686ms	479ms
	V	114ms	959ms	380ms
	S	219kB	2,127kB	3,456kB
p610	P	3,175ms	2,488ms	989ms
	V	472ms	2614ms	818ms
	S	517kB	4,084kB	7,607kB
p751	P	3,882ms	1,951ms	2,131ms
	V	824ms	6407ms	1,793ms
	S	828kB	6,394kB	15,104kB

Table 6. Table of results comparing several generic proof systems operating over $\mathbb{F}_{p^2}$ for the R1CS instantiation of $\mathcal{R}_{2^k\text{-MP}}$ without relaxations. The soundness/zero-knowledge security level is set according to Tab. 5 and P, V, S correspond to proof time, verification time, and proof size respectively. Results displayed are for single-threaded performance.

Parameter		Aurora	Ligero	Limbo
p441+	P	1,216ms	427ms	330ms
	V	98ms	493ms	264ms
	S	166kB	1,733kB	3,496kB
p509+	P	1,440ms	537ms	438ms
	V	120ms	603ms	342ms
	S	182kB	1,967kB	4,657kB
p619+	P	2,287ms	1,130ms	922ms
	V	239ms	849ms	746ms
	S	338kB	2,414kB	10,327kB
p761+	P	3,030ms	1,044ms	1,938ms
	V	431ms	1,951ms	1,594ms
	S	551kB	4,004kB	20,588kB

Table 7. Table of results comparing generic proof systems operating over $\mathbb{F}_p$ for the projected R1CS instantiation of $R_{2^k\text{-MP}}$ operating over fields with characteristic of the form $2^a 3^b f + 1$. Soundness/zero-knowledge security levels set according to Tab. 5. Results displayed are for single threaded performance.

5 Conclusion

In conclusion, we show that generic proof systems are competitive when applied to isogeny-based relations, by giving a proof of concept for an isogeny proof of knowledge using a compact R1CS instance, whose security is based on the underlying proof systems. Our best experimental result shows an order of magnitude improvement for prover and verifier time compared to the state-of-the-art tailor-made isogeny protocol, SECUER PoK.

A remark on signatures. Several post-quantum signature schemes have been proposed by applying MPCitH proof systems to PRFs, such as [dDOS19,ZCD$^+$20, Bd20, BdK$^+$21]. The approach follows one of two processes, given a uniform secret key k:

1. The public key is y such that $f(k) = y$ for a one-way function f. A signature corresponds to a non-interactive proof that *"I know a k such that $f(k) = y$"* where the message m is incorporated into the randomness of the challenges.
2. The public key is $\mathrm{PRF}_k(0^\lambda)$, and a signature is then an evaluation of $\mathrm{PRF}_k(m)$ attached with a proof that *"I know a k such that I can compute both $PRF_k(m)$ and $PRF_k(0^\lambda)$"*.

Given a secure PRF, the latter approach is somewhat agnostic to the proof system in question. However, in the former case, it is unclear that proofs obtained from the BCS transform applied to IOPs can yield a secure signature scheme analogous to Fiat-Shamir applied to Σ-protocols. Some works [FKMV12, GKK$^+$22] indicate the non-malleability or *simulation extractability* is an important notion in the security of this construction. Simulation extractability provides that a malicious prover cannot forge a valid proof without knowledge of the witness, even after seeing polynomially many valid proofs. In particular, this notion seems to yield a direct reduction to EUF-CMA. To this date, the security of the BCS transform with messages incorporated into the verifier's randomness lacks sufficient analysis, and it is unclear as to what property is necessary and sufficient in order to construct signatures by (1). If this is achieved, it is straightforward to convert the isogeny proof of knowledge into a signature scheme based on the hardness isogeny path-finding, where the one-way function is essentially the CGL hash function.

Acknowledgements

This work is a result of a collaboration at the University of Auckland, and is supported by the Ministry for Business, Innovation and Employment of New Zealand; the Cyber Security Research Flanders with reference number VR20192203; the Defense Advanced Research Projects Agency (DARPA); and the Space and Naval Warfare Systems Center, Pacific (SSC Pacific) under contract No. FA8750-19-C-0502.

Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of any of the funders. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation therein.

We would like to acknowledge and thank Steven Galbraith for sharing his valuable supervision and insight on the project; as well as Cyprien Delpech de Saint Guilhem and Titouan Tanguy for their support with the Limbo implementation. We thank Luca de Feo for sharing his valuable suggestions and opinions to improve the clarity and correctness of this work.

References

ABCP22. Shahla Atapoor, Karim Baghery, Daniele Cozzo, and Robi Pedersen. Csi-shark: Csi-fish with sharing-friendly keys. Cryptology ePrint Archive, Paper 2022/1189, 2022. https://eprint.iacr.org/2022/1189.

ACC$^+$17. R. Azarderakhsh, M. Campagna, C. Costello, L. De Feo, B. Hess, A. Jalali, D. Jao, B. Koziel, B. LaMacchia, and et al. Longa, P. Supersingular isogeny key encapsulation (SIKE). Submission to the NIST Post-Quantum Standardization Project, 2017. https://csrc.nist.gov/csrc/media/Projects/post-quantum-cryptography/documents/round-4/submissions/SIKE-spec.pdf.

ACC$^+$19. Gora Adj, Daniel Cervantes-Vázquez, Jesús-Javier Chi-Domínguez, Alfred Menezes, and Francisco Rodríguez-Henríquez. On the cost of computing isogenies between supersingular elliptic curves. In Carlos Cid and Michael J. Jacobson Jr:, editors, *SAC 2018*, volume 11349 of *LNCS*, pages 322–343. Springer, Heidelberg, August 2019.

AEK$^+$22. Michel Abdalla, Thorsten Eisenhofer, Eike Kiltz, Sabrina Kunzweiler, and Doreen Riepel. Password-authenticated key exchange from group actions. Cryptology ePrint Archive, Paper 2022/770, 2022. https://eprint.iacr.org/2022/770.

AHIV17. Scott Ames, Carmit Hazay, Yuval Ishai, and Muthuramakrishnan Venkitasubramaniam. Ligero: Lightweight sublinear arguments without a trusted setup. In Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu, editors, *ACM CCS 2017*, pages 2087–2104. ACM Press, October / November 2017.

BBD$^+$22. Jeremy Booher, Ross Bowden, Javad Doliskani, Tako Boris Fouotsa, Steven D. Galbraith, Sabrina Kunzweiler, Simon-Philipp Merz, Christophe Petit, Benjamin Smith, Katherine E. Stange, Yan Bo Ti, Christelle Vincent, José Felipe Voloch, Charlotte Weitkämper, and Lukas Zobernig. Failing to hash into supersingular isogeny graphs. Cryptology ePrint Archive, Report 2022/518, 2022. https://eprint.iacr.org/2022/518.

BBHR18. Eli Ben-Sasson, Iddo Bentov, Yinon Horesh, and Michael Riabzev. Fast reed-solomon interactive oracle proofs of proximity. In Ioannis Chatzigiannakis, Christos Kaklamanis, Dániel Marx, and Donald Sannella, editors, *ICALP 2018*, volume 107 of *LIPIcs*, pages 14:1–14:17. Schloss Dagstuhl, July 2018.

BCC+16. Jonathan Bootle, Andrea Cerulli, Pyrros Chaidos, Jens Groth, and Christophe Petit. Efficient zero-knowledge arguments for arithmetic circuits in the discrete log setting. In Marc Fischlin and Jean-Sébastien Coron, editors, *EUROCRYPT 2016, Part II*, volume 9666 of *LNCS*, pages 327–357. Springer, Heidelberg, May 2016.

BCC+22. Andrea Basso, Giulio Codogni, Deirdre Connolly, Luca De Feo, Tako Boris Fouotsa, Guido Maria Lido, Travis Morrison, Lorenz Panny, Sikhar Patranabis, and Benjamin Wesolowski. Supersingular curves you can trust. Cryptology ePrint Archive, Paper 2022/1469, 2022. `https://eprint.iacr.org/2022/1469`.

BCG+14. Eli Ben-Sasson, Alessandro Chiesa, Christina Garman, Matthew Green, Ian Miers, Eran Tromer, and Madars Virza. Zerocash: Decentralized anonymous payments from bitcoin. In *2014 IEEE Symposium on Security and Privacy*, pages 459–474. IEEE Computer Society Press, May 2014.

BCR+19. Eli Ben-Sasson, Alessandro Chiesa, Michael Riabzev, Nicholas Spooner, Madars Virza, and Nicholas P. Ward. Aurora: Transparent succinct arguments for R1CS. In Yuval Ishai and Vincent Rijmen, editors, *EUROCRYPT 2019, Part I*, volume 11476 of *LNCS*, pages 103–128. Springer, Heidelberg, May 2019.

BCS16. Eli Ben-Sasson, Alessandro Chiesa, and Nicholas Spooner. Interactive oracle proofs. In Martin Hirt and Adam D. Smith, editors, *TCC 2016-B, Part II*, volume 9986 of *LNCS*, pages 31–60. Springer, Heidelberg, October / November 2016.

Bd20. Ward Beullens and Cyprien de Saint Guilhem. LegRoast: Efficient post-quantum signatures from the Legendre PRF. In Jintai Ding and Jean-Pierre Tillich, editors, *Post-Quantum Cryptography - 11th International Conference, PQCrypto 2020*, pages 130–150. Springer, Heidelberg, 2020.

BD21. Jeffrey Burdges and Luca De Feo. Delay encryption. In Anne Canteaut and François-Xavier Standaert, editors, *EUROCRYPT 2021, Part I*, volume 12696 of *LNCS*, pages 302–326. Springer, Heidelberg, October 2021.

BdK+21. Carsten Baum, Cyprien de Saint Guilhem, Daniel Kales, Emmanuela Orsini, Peter Scholl, and Greg Zaverucha. Banquet: Short and fast signatures from AES. In Juan Garay, editor, *PKC 2021, Part I*, volume 12710 of *LNCS*, pages 266–297. Springer, Heidelberg, May 2021.

BDK+22. Ward Beullens, Samuel Dobson, Shuichi Katsumata, Yi-Fu Lai, and Federico Pintore. Group signatures and more from isogenies and lattices: Generic, simple, and efficient. In Orr Dunkelman and Stefan Dziembowski, editors, *EUROCRYPT 2022, Part II*, volume 13276 of *LNCS*, pages 95–126. Springer, Heidelberg, May / June 2022.

BFH+20. Rishabh Bhadauria, Zhiyong Fang, Carmit Hazay, Muthuramakrishnan Venkitasubramaniam, Tiancheng Xie, and Yupeng Zhang. Ligero++: A new optimized sublinear IOP. In Jay Ligatti, Xinming Ou, Jonathan Katz, and Giovanni Vigna, editors, *ACM CCS 2020*, pages 2025–2038. ACM Press, November 2020.

BKP20. Ward Beullens, Shuichi Katsumata, and Federico Pintore. Calamari and Falafl: Logarithmic (linkable) ring signatures from isogenies and lattices. In Shiho Moriai and Huaxiong Wang, editors, *ASIACRYPT 2020, Part II*, volume 12492 of *LNCS*, pages 464–492. Springer, Heidelberg, December 2020.

BKV19. Ward Beullens, Thorsten Kleinjung, and Frederik Vercauteren. CSI-FiSh: Efficient isogeny based signatures through class group computations. In Steven D. Galbraith and Shiho Moriai, editors, *ASIACRYPT 2019, Part I*, volume 11921 of *LNCS*, pages 227–247. Springer, Heidelberg, December 2019.

Brö08. Reinier Bröker. Constructing elliptic curves of prescribed order. 2008.

BS22. Ward Beullens and Gregor Seiler. Labrador: Compact proofs for r1cs from module-sis. Cryptology ePrint Archive, Paper 2022/1341, 2022. `https://eprint.iacr.org/2022/1341`.

CD22. Wouter Castryck and Thomas Decru. An efficient key recovery attack on sidh (preliminary version). Cryptology ePrint Archive, Paper 2022/975, 2022. `https://eprint.iacr.org/2022/975`.

CLG09. Denis Xavier Charles, Kristin E. Lauter, and Eyal Z. Goren. Cryptographic hash functions from expander graphs. *Journal of Cryptology*, 22(1):93–113, January 2009.

CLM$^+$18. Wouter Castryck, Tanja Lange, Chloe Martindale, Lorenz Panny, and Joost Renes. CSIDH: An efficient post-quantum commutative group action. In Thomas Peyrin and Steven Galbraith, editors, *ASIACRYPT 2018, Part III*, volume 11274 of *LNCS*, pages 395–427. Springer, Heidelberg, December 2018.

CLN16. Craig Costello, Patrick Longa, and Michael Naehrig. Efficient algorithms for supersingular isogeny Diffie-Hellman. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part I*, volume 9814 of *LNCS*, pages 572–601. Springer, Heidelberg, August 2016.

DDGZ21. Luca De Feo, Samuel Dobson, Steven D. Galbraith, and Lukas Zobernig. SIDH proof of knowledge. Cryptology ePrint Archive, Report 2021/1023, 2021. `https://eprint.iacr.org/2021/1023`.

dDOS19. Cyprien de Saint Guilhem, Lauren De Meyer, Emmanuela Orsini, and Nigel P. Smart. BBQ: Using AES in picnic signatures. In Kenneth G. Paterson and Douglas Stebila, editors, *SAC 2019*, volume 11959 of *LNCS*, pages 669–692. Springer, Heidelberg, August 2019.

DFJP14. Luca De Feo, David Jao, and Jérôme Plût. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. *Journal of Mathematical Cryptology*, 8(3):209–247, 2014.

DJP11. Luca De Feo, David Jao, and Jérôme Plût. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. Cryptology ePrint Archive, Report 2011/506, 2011. `https://eprint.iacr.org/2011/506`.

DKL$^+$20. Luca De Feo, David Kohel, Antonin Leroux, Christophe Petit, and Benjamin Wesolowski. SQISign: Compact post-quantum signatures from quaternions and isogenies. In Shiho Moriai and Huaxiong Wang, editors, *ASIACRYPT 2020, Part I*, volume 12491 of *LNCS*, pages 64–93. Springer, Heidelberg, December 2020.

dOT21. Cyprien de Saint Guilhem, Emmanuela Orsini, and Titouan Tanguy. Limbo: Efficient zero-knowledge MPCitH-based arguments. In Giovanni Vigna and Elaine Shi, editors, *ACM CCS 2021*, pages 3022–3036. ACM Press, November 2021.

DPB17. Javad Doliskani, Geovandro C. C. F. Pereira, and Paulo S. L. M. Barreto. Faster cryptographic hash function from supersingular isogeny graphs. Cryptology ePrint Archive, Paper 2017/1202, 2017. `https://eprint.iacr.org/2017/1202`.

FJR22. Thibauld Feneuil, Antoine Joux, and Matthieu Rivain. Syndrome decoding in the head: Shorter signatures from zero-knowledge proofs. Cryptology ePrint Archive, Report 2022/188, 2022. `https://eprint.iacr.org/2022/188`.

FKMV12. Sebastian Faust, Markulf Kohlweiss, Giorgia Azzurra Marson, and Daniele Venturi. On the non-malleability of the Fiat-Shamir transform. In Steven D. Galbraith and Mridul Nandi, editors, *INDOCRYPT 2012*, volume 7668 of *LNCS*, pages 60–79. Springer, Heidelberg, December 2012.

FMRV22. Thibauld Feneuil, Jules Maire, Matthieu Rivain, and Damien Vergnaud. Zero-knowledge protocols for the subset sum problem from MPC-in-the-head with rejection. Cryptology ePrint Archive, Report 2022/223, 2022. `https://eprint.iacr.org/2022/223`.

FS87. Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In Andrew M. Odlyzko, editor, *CRYPTO'86*, volume 263 of *LNCS*, pages 186–194. Springer, Heidelberg, August 1987.

Gal99. Steven D. Galbraith. Constructing isogenies between elliptic curves over finite fields. *LMS Journal of Computation and Mathematics*, 2:118–138, 1999.

GKK^{+}22. Chaya Ganesh, Hamidreza Khoshakhlagh, Markulf Kohlweiss, Anca Nitulescu, and Michał Zając. What Makes Fiat–Shamir zkSNARKs (Updatable SRS) Simulation Extractable? In Clemente Galdi and Stanislaw Jarecki, editors, *Security and Cryptography for Networks*, pages 735–760, Cham, 2022. Springer International Publishing.

GKPV21. Wissam Ghantous, Shuichi Katsumata, Federico Pintore, and Mattia Veroni. Collisions in supersingular isogeny graphs and the sidh-based identification protocol. Cryptology ePrint Archive, Paper 2021/1051, 2021. `https://eprint.iacr.org/2021/1051`.

GLS^{+}21. Alexander Golovnev, Jonathan Lee, Srinath Setty, Justin Thaler, and Riad S. Wahby. Brakedown: Linear-time and post-quantum SNARKs for R1CS. Cryptology ePrint Archive, Report 2021/1043, 2021. `https://eprint.iacr.org/2021/1043`.

GMNO18. Rosario Gennaro, Michele Minelli, Anca Nitulescu, and Michele Orrù. Lattice-based zk-SNARKs from square span programs. In David Lie, Mohammad Mannan, Michael Backes, and XiaoFeng Wang, editors, *ACM CCS 2018*, pages 556–573. ACM Press, October 2018.

GPV21. Wissam Ghantous, Federico Pintore, and Mattia Veroni. Collisions in supersingular isogeny graphs and the SIDH-based identification protocol. Cryptology ePrint Archive, Report 2021/1051, 2021. `https://eprint.iacr.org/2021/1051`.

IKOS07. Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. Zero-knowledge from secure multiparty computation. In David S. Johnson and Uriel Feige, editors, *39th ACM STOC*, pages 21–30. ACM Press, June 2007.

IKOS09. Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. Zero-knowledge proofs from secure multiparty computation. *SIAM Journal on Computing*, 39(3):1121–1152, 2009.

Ish20. Yuval Ishai. Zero-knowledge proofs from information-theoretic proof systems. *Zkproofs Blog*, 2020.

JD11. David Jao and Luca De Feo. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. In Bo-Yin Yang, editor, *Post-*

Quantum Cryptography - 4th International Workshop, PQCrypto 2011, pages 19–34. Springer, Heidelberg, November / December 2011.

KKW18. Jonathan Katz, Vladimir Kolesnikov, and Xiao Wang. Improved non-interactive zero knowledge with applications to post-quantum signatures. In David Lie, Mohammad Mannan, Michael Backes, and XiaoFeng Wang, editors, *ACM CCS 2018*, pages 525–537. ACM Press, October 2018.

Ler21. Antonin Leroux. A new isogeny representation and applications to cryptography. Cryptology ePrint Archive, Report 2021/1600, 2021. `https://eprint.iacr.org/2021/1600`.

LGd21. Yi-Fu Lai, Steven D. Galbraith, and Cyprien de Saint Guilhem. Compact, efficient and UC-secure isogeny-based oblivious transfer. In Anne Canteaut and François-Xavier Standaert, editors, *EUROCRYPT 2021, Part I*, volume 12696 of *LNCS*, pages 213–241. Springer, Heidelberg, October 2021.

MMP22. Marzio Mula, Nadir Murru, and Federico Pintore. Random sampling of supersingular elliptic curves. Cryptology ePrint Archive, Report 2022/528, 2022. `https://eprint.iacr.org/2022/528`.

Onu21. Hiroshi Onuki. On oriented supersingular elliptic curves. *Finite Fields and Their Applications*, 69:101777, 2021.

Piz90. Arnold K. Pizer. Ramanujan graphs and Hecke operators. *Bulletin of the American Mathematical Society*, 23(1):127–137, 1990.

Rob22. Damien Robert. Breaking sidh in polynomial time. Cryptology ePrint Archive, Paper 2022/1038, 2022. `https://eprint.iacr.org/2022/1038`.

Sil09. Joseph H. Silverman. *The Arithmetic of Elliptic Curves*, volume 106 of *Graduate Texts in Mathematics*. Springer New York, New York, NY, 2009.

Ste22. Bruno Sterner. Commitment Schemes from Supersingular Elliptic Curve Isogeny Graphs. *Mathematical Cryptology*, 1(2):40–51, March 2022.

Tha20. Justin Thaler. Proofs, arguments, and zero-knowledge, 2020.

vW99. Paul C. van Oorschot and Michael J. Wiener. Parallel collision search with cryptanalytic applications. *Journal of Cryptology*, 12(1):1–28, January 1999.

Was08. Lawrence C Washington. *Elliptic curves: number theory and cryptography*. Chapman and Hall/CRC, 2008.

XZS22. Tiancheng Xie, Yupeng Zhang, and Dawn Song. Orion: Zero knowledge proof with linear prover time. Cryptology ePrint Archive, Paper 2022/1010, 2022. `https://eprint.iacr.org/2022/1010`.

ZCD$^+$20. Greg Zaverucha, Melissa Chase, David Derler, Steven Goldfeder, Claudio Orlandi, Sebastian Ramacher, Christian Rechberger, Daniel Slamanig, Jonathan Katz, Xiao Wang, Vladmir Kolesnikov, and Daniel Kales. Picnic. Technical report, National Institute of Standards and Technology, 2020. available at `https://csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions`.

ZXZS19. Jiaheng Zhang, Tiancheng Xie, Yupeng Zhang, and Dawn Song. Transparent polynomial delegation and its applications to zero knowledge proof. Cryptology ePrint Archive, Report 2019/1482, 2019. `https://eprint.iacr.org/2019/1482`.

A Preventing Backtracking

In some applications, one might want a guarantee that the isogeny proven is cyclic, which in our setting, is equivalent to showing that the isogeny walk is non-backtracking. That is, a walk which does not immediately traverse the same edge twice.

Theorem 2. *An isogeny $\phi : E_0 \to E_k$ of degree 2^k is cyclic if and only if ϕ's decomposition into 2-isogenies as a walk on the supersingular isogeny graph is non-backtracking.*

Proof. See [CLG09, Prop. 1] [DPB17, Prop. 3.2]

In the modular polynomial relation we introduce, we do not provide any guarantee that our isogeny is non-backtracking (and hence cyclic). However, with minor overhead, it is possible to add this requirement. Observe that, given an isogeny walk from E_0 to E_k of length k, with a j-invariant sequence $j_0, ..., j_k$, a backtracking walk implies that there exists an $i \in \{1, ..., k-1\}$ such that $j_{i-1} = j_{i+1}$. So it suffices to show that

$$\delta_i = j_{i-1} - j_{i+1} \neq 0 \text{ for all } i \in \{1, ..., k-1\}.$$

One can realise inequality in an arithmetic circuit with the following process: given two numbers a, b, we may show that they are distinct if and only if there exists an inverse of $(a - b)$ over the field. Alternatively, there exists c such that $(a - b) \cdot c = 1$. The inverse $c := (a - b)^{-1}$ can be precomputed by the prover and given as a part of the input.

We can perform an additional optimization step to minimise the number of precomputed inverses for the prover, the calculation of which is expensive. Indeed, the prover can accumulate the product of the difference terms δ_i, and check that the product is nonzero. In particular, our resulting conditions to prevent backtracking are that:

1. Compute $\delta = \prod \delta_i = \prod_{i=1}^{k-1} (j_{i-1} - j_{i+1})$.
2. Input δ' such that $\delta\delta' = 1$,

where the δ term will be non-zero if and only if all δ_i are non-zero, which is true if (but not only if) the walk is non-backtracking. We note that this check will also prevent the use of 2-cycles (with two distinct edges), which may be cyclic, but are of little consequence in practice.

It is straightforward to add these constraints to our previous R1CS instance. In the $\mathbb{F}_{p^2}$ setting, this would add an additional $k-1$ constraints and variables for the product check; and one constraint and variable (the inverse given as input) for the inverse check. This version yields $5k + 3$ variables and $5k + 2$ constraints in total, which means only a 25% overhead if compared to the original instance size. We expect this to subject only a minor performance cost, as the protocol scales well with instance size (as seen in the difference between Sec. 4.2 and Sec. 4.1).