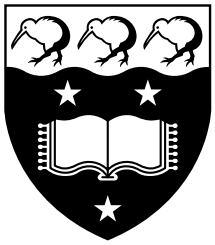
# Constructing Secure Protocols from Proofs of Knowledge and Isogenies



Shai Levin

Department of Mathematics
The University of Auckland

Supervisor: Steven Galbraith
Co-supervisor: Jeroen Schillewaert

A thesis submitted in partial fulfilment of the requirements for the degree of PhD in Mathematics, The University of Auckland, 2025.

# Abstract

Constructing efficient isogeny-based protocols is an active area of research in post-quantum cryptography. An important building block in constructing such protocols is non-interactive zero-knowledge proofs of knowledge, which convince a verifier that a prover possesses some secret information about an isogeny without revealing it.

Prior proofs of knowledge of an isogeny based on $\Sigma$-protocols [JD11, DDGZ22, GPS17, BCC+23] suffer from various deficiencies. They require computing large degree coprime $N'$-isogenies, which means they must either work over larger field extensions or choose a prime large enough to have rational $N'$-torsion. Further, they suffer from poor performance due to their small challenge space, resulting in many parallel repetitions to achieve negligible soundness error.

This thesis presents new techniques for constructing isogeny proofs of knowledge by constructing generic instances which encode isogeny relations. These instances can then be implemented with a generic proof system (i.e. a zk-SNARK) to produce non-interactive arguments for isogeny relations, such as proving knowledge of a cyclic $2^n$-isogeny. Compared to prior approaches, zk-SNARKs allow for several orders of magnitude improvement to prover and verification times, and proofs are additionally *succinct*, they scale sublinearly with the witness size. To motivate our approach with an application, we construct a variant of the CGL hash function [CLG09], requiring trusted setup, along with an associated proof of honest evaluation via generic techniques. Given the function satisfies a conjectured notion of *unpredictability*, we use it, along with its evaluation proof to construct a *verifiable random function* (VRF), in the random oracle model. As an independent contribution, we prove the security of the associated generic VRF transformation.

As an orthogonal contribution, we explore the shortcomings of various works which construct proofs of knowledge in modern literature in a dedicated cryptanalysis section. In particular, we show that a peer-reviewed variant of SeaSign [Kim24] is not zero-knowledge, leading to a key recovery attack; that a proof of knowledge of a commitment to an elliptic curve discrete logarithm [FLM22] is not sound (and propose a fixed protocol), and that the CROSS identification protocol [BBB+24] does not satisfy their claimed level of zero-knowledge.

# Acknowledgements

I have been very fortunate to have had two great mentors during my time as a PhD student. One inside the office, and one inside the ring.

First and foremost, I would like to sincerely thank my supervisor, Steven Galbraith, for his guidance and support throughout my PhD. Steven was always willing to schedule a meeting with short notice when the work load was high, and on the other hand, take the time for an hour of chit-chat when we were stumped on ideas. Steven was also supportive of my participation in my sport – Muay Thai – particularly when I wanted to take a short notice fight for a New Zealand national Muay Thai title, 11 days before my thesis submission, because he knew what it meant to me and that my thesis was in good shape[1].

One of the unintentional consequences of having such a strong supervisor over the course of my PhD has been developing an internal model of intuition. I have the sense that over the past three years, I have developed a simulacrum of Steven's perspective on my own, and other's work in the field – "that security proof needs more details", or "this is terrible notation", or perhaps most notably – "that seems very fishy". This has been a double-edged sword, as I have often found myself in the position of having to convincing my internal, simulated Steven that my work is actually not crap. But it has also given me the confidence to pursue my own ideas, and to trust "my own" intuition. For this I am immensely grateful.

I would also like to thank my co-supervisor, Jeroen Schillewaert, for his cheerful quips in the office hallway and many conversations over a beer.

Outside of the office, I am immensely grateful for my second home, City Lee Gar – and the coaches and sparring partners I had the privilege of learning from. Of particular note, I would like to thank my second mentor, my coach, Tony Angelov. Tony is absolutely unflappable. I have never seen him anything but completely relaxed and focused on the task at hand, even under immense pressure. He has taught me that one of the strongest traits in a leader, and a fighter, is to be open, understand the true nature behind people's actions, and to have heart.

[1] I won, by the way.

Lastly, I would like to thank my family, who live in South Africa and have always supported me in my endeavours: my mother, Terri, who I could spend hours talking to about anything and everything; my father, Alan, who is responsible for my fervent ambition; and my siblings, Chiara and Ethan, who give me a reason to be an oldest brother.

# Contents

# Chapter 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 strengthens the assumption by giving additional torsion point information; CSIDH based on group actions [CLM+18]; SQI-Sign, a signature scheme based on endomorphism rings [DKL+20]; its many variants which rely on new tools for computing evaluation of large prime degree isogenies via higher dimensional representations [BDD+24, NOC+24, DF24, DLRW24], and others, such as [FMP23, Bas24a, Bas24b, BMP23, Ste22]. Even though there was a recent cryptanalytic breakthrough on SIDH [MMP+23, CD23, Rob23], other cryptosystems (including those cited above) remain unaffected. Additionally, a variety of advanced schemes and protocols based on isogenies, such as oblivious transfer and exotic signatures, have been proposed in the literature, such as [BKV19, BKP20, LGD21, BD21, BDK+22].

A recent development in isogeny-based cryptography has been the proposal of several *verifiable random functions* (VRFs) [Lai24, Ler25]. VRFs are a cryptographic primitive which allows an evaluator of a pseudorandom function to prove the correctness of their output. A VRF is instantiated with a public-private key pair $(\mathsf{pk}, \mathsf{sk})$, and on the input of a message $m$ and private key $\mathsf{sk}$, outputs a pseudorandom value $h$ and proof $\pi$. A verifier may then take $(\mathsf{pk}, m, h, \pi)$ and accept or reject. VRFs must satisfy *residual pseudorandomness* (a related but distinct notion compared to the *pseudorandomness* of a PRF) which states that for a new message $m$, the output $h$ is indistinguishable from random, even if an adversary has already seen evaluations and proofs for arbitrary, distinct messages. VRFs also satisfy *unique provability*, which means that two distinct evaluations of the VRF under the same message and secret key cannot both have accepting proofs.

First introduced in [MRV99], VRFs have found applications in blockchain consensus,

such as in Algorand [CM19], where a VRF is used after each block in a *lottery* to determine who forms the next block; distributed randomness beacons [CMB23], where publicly generated randomness can be generated in a distributed, trust-less computation; and DNSSEC [GNP+15].

Until recently, VRF constructions have been based exclusively on classical assumptions which require the hardness of computing discrete-logarithms or integer factoring, and thus are not post-quantum secure. Hence, it is of interest to propose candidates for post-quantum VRFs. Post-quantum secure VRFs have been proposed from lattice [EKS+21, ESLR23] and hash [BDE+22, EEK+23] assumptions. We note that the constructions in [EKS+21, EEK+23] are *short-term* VRFs, which trade off long-term usability for higher efficiency. Another short-term VRF has been proposed in [BDE+22], but was broken in [BSN24]. We briefly discuss the isogeny-based VRFs below. Lai's construction is based on a Naor-Reingold type PRF [NR99], hardness is based on decisional Diffie-Hellman (DDH) in the effective group action setting [ADMP20] and Lai introduces new proof systems to realise verifiability. In contrast, Leroux's VRF evaluations are large prime degree isogenies computed via the Deuring correspondence and proven correct by providing a higher dimensional representation of that isogeny. Leroux's VRF relies on a one-more type computational assumption, and results in the smallest proof sizes in the post-quantum setting (cf. [Ler25, Table 1]). It is worth noting that post-quantum VRFs in the literature either suffer from poor efficiency, or rely on novel computational assumptions – hence there is a trade-off between minimal assumptions and efficiency.

Returning to the wider context of isogeny cryptosystems, every isogeny computation starts 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 elliptic curve without knowing its endomorphism ring [BBD+22, MMP22], is currently a computationally infeasible task, 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+23] 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, ABCP23, Lai24]. 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 prior state-of-the-art works based on $\Sigma$-protocols are given by [JD11, DDGZ22, GPS17, BCC+23], yet there is still room for improvement. Suppose 300 participants run the ceremony single-threaded on a normal machine[1], the protocol will take roughly an hour to complete for $\lambda = 128$, and 13 hours for $\lambda = 256$. Other protocols would also benefit from zero-knowledge proofs for other, more tailored isogeny relations. In some cases, these proofs are not believed to be possible with existing techniques via $\Sigma$-protocols. One example of which is [Bas24a], the initial preprint publication of which included a split-KEM and OPRF construction, which has been removed since the security of these constructions would require a proof of honest evaluation of a secret degree isogeny, which is not known to be possible via $\Sigma$-protocols.

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 zk-SNARKs[2], which allow a prover to prove or argue the knowledge of any $\mathsf{NP}$ relation, have advanced the field significantly in recent years. zk-SNARKs allow 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 – see [Tha23] for a recent survey. To quote from the preface of this survey:

> *"A major benefit of taking 7 years to complete this manuscript is the many exciting developments that can now be included. This survey would have looked very different had it been completed in 2015, or even in 2020 (over 1/3 of the content covered did not exist 7 years ago). During this period, the various approaches to the design of zero-knowledge arguments, and the relationships between them, have come into finer focus. Yet owing to the sheer volume of research papers, it is increasingly challenging for those first entering the area to extract a clear picture from the literature itself."*

Indeed, the area continues to develop, even since 2023, and as of the time of writing this thesis in early 2025, there are many new candidates for the post-quantum state-of-the-art [XZS22, ZCF24, BFK+24, GLH+25, ACFY25]. Several of these SNARKs are also *field-agnostic*, which operate using expander codes and do not impose any restrictions on the underlying finite field. Additionally, several techniques have been

[1] Specifically, an ARM Apple M1 Pro

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

introduced which provide a blueprint to proving stronger soundness guarantees of the non-interactive proof systems [KPT23, FFK+23]. In general, these generic proof systems work well with symmetric primitives and have applications in post-quantum cryptosystems [ZCD+20, GMNO18, DDOS19, BDK+21], and privacy-preserving blockchain protocols such as [BCG+14]. Due to their flexibility, protocols can benefit from the advances in proof systems without major changes to the underlying protocol (simply by changing the proof system, provided they support the same security guarantees and language of computation).

Applying generic proof systems to isogeny-based cryptography remains uncommon. Though there exists a verifiable delay function from isogenies using a SNARG[3] [CSRT22], 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 Contributions

The primary contribution of this thesis is to answer the question above affirmatively. We first introduce the relevant preliminaries and notation in Chapter 2. Along the way, and as a warm up, we briefly explore the use of Σ-protocols to construct proofs for generic relations on Pedersen commitments over elliptic curve groups. In the final chapter, we cryptanalyse various protocols based on Σ-protocols, showing that the security properties of which are often misunderstood or improperly stated. In some cases, such as with [Kim24], this can be disastrous, leading to a key-recovery attack, whereas in others [BBB+24], the security properties are simply not as strong as previously thought.

**Proving arithmetic statements on Pedersen commitments** In Chapter 3, we collect and present various Σ-protocols for proving arithmetic statements on Pedersen values, which are either obtained from [WTs+18, Sch91] or appear as folklore construc-

[3] succinct, non-interactive argument

tions. In particular, given a Pedersen commitment $C = g^x h^r = \mathsf{Commit}(x; r)$, we show how to prove:

- **Knowledge of an opening (Section 3.1):** Given $C$, knowledge of $x$ and $r$ such that $C = g^x h^r$.
- **Equality of Pedersen commitments (Section 3.2):** Given $C_1, C_2$, knowledge of $x, r_1, r_2$ such that $C_1 = g^x h^{r_1}$ and $C_2 = g^x h^{r_2}$.
- **Multiplicative relation of Pedersen commitments (Section 3.3):** Given $C_1, C_2, C_3$, knowledge of $x_1, x_2, r_1, r_2, r_3$ such that $C_1 = g^{x_1} h^{r_1}$, $C_2 = g^{x_2} h^{r_2}$ and $C_3 = g^{x_1 \cdot x_2} h^{r_3}$.
- **Commitments opens to a non-zero value (Section 3.4):** Given $C$, knowledge of $x$ and $r$ such that $C = g^x h^r$ and $x \neq 0$.

We then show how to use these proofs to build a generic proof system for Rank 1 Constraint Systems (R1CS) using Pedersen commitments and $\Sigma$-protocols in Section 3.5. R1CS is a widely used relation for expressing arbitrary arithmetic statements, which we use in the following chapters.

**Proving isogeny relations with generic proof systems** In Chapter 4, we show how to construct efficient proofs of knowledge for three isogeny relations using generic proof systems. In particular, in Sections 4.1 and 4.2, we show how to prove:

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

via R1CS instances which encode the language of $\ell^k$-isogenies, but we also extend this to the case that $\phi$ is cyclic. We include the original approach of [CLL23] as a part of this thesis, but then introduce an improved approach first described in [LP25] via radical isogenies [CDV20]. We focus on the case where $\ell = 2$, but the approach extends to arbitrary $\ell$ provided the existence of radical isogeny formula for degree $\ell$. We include a rough comparison of the advantages of our generic approach in Table 1.1. We note that this is not directly an apples-to-apples comparison, since $\Sigma$-protocols require running the protocol over a larger field. We also believe that we can do better by using a more efficient FFT-based proof system over $\mathbb{F}_{p^2}$, such as WHIR [ACFY25].

Next, also originally appearing in the work of [LP25] and included in this thesis, in Section 4.4 we introduce a novel variant of the CGL hash function [CLG09] using radical isogenies (we call this $\mathsf{CGL}$) and we show how to construct R1CS instances for the following relation:

$$\mathcal{R}_{\mathsf{CGL}} = \{((E_0, E_n); m) \mid E_n = \mathsf{CGL}(E_0, m)\} \ ,$$

| | $\log p$ | Prov. Time (ms) | Verif. Time (ms) | Proof Size (kB) |
|---|---|---|---|---|
| [BCC+23] | 434 | 18,150 | 1,930 | 191.19 |
| Ours (estimated) | 256 | 45 | 20 | 320 |

Table 1.1: Performance of our radical R1CS over $\mathbb{F}_p$ for proving knowledge of a cyclic $2^{705}$-isogeny when applied to the protocol of [BFK+24], compared to the performance of the $\Sigma$-protocol of [BCC+23]. We obtain our estimates based on the performance of [BFK+24] on instances of size $< 2^{12}$, since $5 \cdot 705 \approx 2^{11.79}$.

Lastly, looking ahead to our VRF construction, in Section 4.5, we construct R1CS instances for the relation:

$$\mathcal{R}_{\mathsf{CGL}/\!/} = \{((E_1, E_2, F_1, F_2); m) \mid E_2 = \mathsf{CGL}(E_1, m) \wedge F_2 = \mathsf{CGL}(F_1, m)\} \ .$$

These R1CS, when applied to an appropriate proof system, yield efficient proofs of knowledge for the above relations.

**Isogeny-based VRFs from Unpredictability Assumptions** We start Chapter 5 by introducing *unpredictable functions* in Section 5.1, which are functions satisfying a one-more type computational hardness assumption that we describe in Problem 3. In Section 5.2, we prove that these functions, together with an associated non-interactive zero-knowledge proof of knowledge of honest evaluation, are enough to build a secure VRF in the random oracle model. We go on by instantiating unpredictable functions via our variant of the CGL hash function in Section 5.3. Using the proof system from Section 4.5, that two isogeny walks have been computed using the same input, we can add verifiability to our scheme. This approach, which also appears in [LP25], yields a very fast protocol in terms of evaluation and verification cost. See Table 5.2 for a comparison with other VRF works in the literature. The recent VRF from isogenies proposed by Leroux [Ler25] uses the interplay of the Deuring correspondence with higher dimensional isogenies to prove and verify the correct output. Their protocol is also based on a new one-more type assumption and leads to particularly small proof sizes. In contrast, using [BFK+24] as the underlying proof system, our VRF evaluation time is expected to be much faster than the one in [Ler25], while the verification time is comparable. Furthermore, our protocol enjoys very small public key sizes. On the downside, our proof size is considerably larger, but strongly depends on the SNARK used. While [BFK+24] still yields large proof sizes in our setting, the state-of-the-art has developed significantly even within the past 24 months. We expect that further improvements to expander code multi-linear commitments will continue to improve estimated proof sizes. At present our protocol requires a starting curve of unknown

endomorphism ring, obtained via a trusted setup ceremony such as [BCC+23]. VRFs are typically applied in the context of multi-party protocols, where this ceremony may be performed in the setup phase. However, it is possible that parameters may be constructed such that the protocol may achieve full unique provability without the need for a trusted setup, as discussed in Section 5.3.2.

In Section 5.4, we present an alternative instantiation of our VRF construction from unpredictable functions. This approach was, for comparative reasons, designed to leverage the use of Σ-protocols, rather than generic proofs. Instead of computing isogenies in a bit-by-bit CGL style approach, we compute them via a CGL variant using Vélu's formula as in [DPB24]. To realise verifiability, we introduce a new Σ-protocol that proves two isogenies have been computed using the same input key. This proof can be seen as a extension of the proof of isogeny knowledge introduced in [DDGZ22, Section 3] with challenge space of size 3, and as a (more efficient) simplification of the proofs of parallel isogeny from [BKW20, Bas24b]. This proof requires the use of coprime $N'$-isogenies, as in SIDH[4] [JD11]. While this VRF is not competitive with the CGL and SNARKs design from Section 5.2, we add it to our work as a comparative benchmark to our other instantiation. We further believe that the new proof system is of independent interest for other isogeny-based protocols, where one wants to prove that two isogenies are computed from the same input, even if the isogenies are not necessarily parallel sides of an SIDH square.

**Cryptanalysis of Σ-protocols** In Chapter 6, we investigate various shortcomings in several protocols based on Σ-protocols:

- In Section 6.1, we show that the protocol from [Kim24], a variant of SeaSign [DG19], is not zero-knowledge, and construct a key-recovery attack against the protocol. The content of this section also appears in [Lev25a].

- In Section 6.2, we show that the protocol from [FLM22] does not satisfy 3-special soundness, and hence is not provably sound in its current form. We also discuss a practical attack against its implementation, which only verifies a subset of the parallel repetitions. We go on to construct a fixed version of the protocol which is provably 2-special sound, and zero-knowledge in Section 6.3. The content of these sections also appear in [CLR24].

- In Section 6.4, we show that the NIST additional signatures round 2 candidate, CROSS [BBB+24], does not satisfy its claimed variant of zero-knowledge, and

[4]We emphasise that although we are using the label *SIDH*, our constructions are not susceptible to the SIDH attacks from [CD23, Rob23, MMP+23]. The latter use extra torsion information to recover the secret isogeny, which is not provided as part of our protocols.

that the protocol can at best only satisfy weak computational zero-knowledge. The content of this section also appears in [Lev25b].

# Chapter 2

# Background

## 2.1 Notation and Fundamental Definitions

**Mathematical Notation** We refer to the set $\{1 \dots n\}$ as $[n]$, and the set of integers between $a$ and $b$ (inclusive) as $[a, b]$. We will, on occasion, given an indexed set of vectors, use $\mathbf{v}_i^{(j)}$ to refer to the $i$-th entry of the $j$-th vector.

$\mathbb{N}$ denotes the set of natural numbers, $\mathbb{Z}$ the set of integers, $\mathbb{Q}$ the set of rational numbers, and $\mathbb{R}$ the set of real numbers. $\mathbb{F}$ denotes an arbitrary field, and $\overline{\mathbb{F}}$ its algebraic closure. For a prime power $q$, we denote the finite field of $q$ elements as $\mathbb{F}_q$, and the integers modulo $n$ interchangeably as $\mathbb{Z}/n\mathbb{Z}$ or $\mathbb{Z}_n$. Given a ring (or field) $R$, we refer to its multiplicative group as $R^\times$.

The $n$-adicity of an integer $x$ is the largest power of $n$ that divides $x$. For example, the 2-adicity of 12 is 2.

We use the standard notation for binary operators: $\wedge$ is logical AND, $\vee$ is logical OR, $\neg$ is logical NOT.

We may use $\ell$ and $N$ interchangeably to refer to degrees of isogenies. $\ell$ is typically a small prime, and $N$ is typically a large (cryptographically sized) smooth number (often a prime power).

**Cryptographic Notation** We denote security parameter for cryptographic properties as $\lambda$, which is a natural number. Typically this is fixed across a protocol (for example, a zero-knowledge protocol must satisfy the same security level $\lambda$ for both soundness and zero-knowledge).

An algorithm is a Turing machine. We say that an algorithm is *probabilistic polynomial time* (PPT) if it runs in polynomial time in the security parameter $\lambda$, and may sample random coins from some distribution.

We use $\text{negl}(\cdot)$ to denote a negligible function, and $\text{poly}(\cdot)$ to denote a polynomial function.

In protocol descriptions, we use $a \leftarrow B$ to denote assignment of algorithm $B$'s output to variable $a$, and a conditional equality check $a \stackrel{?}{=} b$ which equals 1 if $a$ is equal to $b$ and 0 otherwise. Given a finite set $S$, we denote the operation of sampling an element $x$ uniformly at random from $S$ as $x \leftarrow\!\!\$\, S$. We occasionally use $a := b$ to denote "defined as", where $a$ is defined as $b$ (similar to the assignment arrow, but also returning the newly defined variable).

We denote random oracles by $O$, which are typically assigned uniformly at random from the set of functions on the respective input/output domains. We denote $A^O$ as PPT algorithm $A$ being given query access to $O$. A PPT algorithm can only make polynomially many queries to $O$. If an algorithm's oracle access is implicit, it may be omitted.

**Perfect Indistinguishability** We say that a pair of probability distributions $X$ and $Y$ are identically distributed, or *perfectly indistinguishable*, if $X = Y$.

**Statistical Indistinguishability** We say that a pair of probability distributions $X$ and $Y$ on a set $S$ are *statistically indistinguishable* with parameter $\lambda$, if the statistical distance (or total variable distance), given by the formula

$$\Delta(X, Y) = \frac{1}{2} \sum_{x \in S} \big| \Pr[x \leftarrow X] - \Pr[x \leftarrow Y] \big|,$$

is negligible $\lambda$.

**Computational Indistinguishability** We say that a pair of probability distributions $X$ and $Y$ are *computationally indistinguishable* with parameter $\lambda$, if for all PPT distinguishers $\mathcal{D}$, the following holds:

$$\big| \Pr[\mathcal{D}(X) = 1] - \Pr[\mathcal{D}(Y) = 1] \big| \leq \text{negl}(\lambda).$$

where $\mathcal{D}$ is given query access to polynomially many samples of $X$ or $Y$.

### 2.1.1 Commitment Schemes

We include the definitions of commitment schemes here as they are relevant to several chapters of this thesis. A commitment scheme is a tuple of algorithms: $\Pi = (\mathsf{SetUp}, \mathsf{Commit}, \mathsf{Open})$ which are defined over a message space $\mathcal{M} \subseteq \{0,1\}^n$ as follows (note that where implicit, we omit the public parameters $\mathsf{pp}$ in the call to the algorithms):

1. $\mathsf{SetUp}(1^\lambda) \to \mathsf{pp}$ which takes as input a security parameter $\lambda$ and outputs public parameters $\mathsf{pp}$ for the commitment scheme. Typically the public parameters define the message space $\mathcal{M}$, commitment space $\mathcal{C}$ and the masking randomness space $\mathcal{N}$.

2. $\mathsf{Commit}(m; \mathsf{pp}) \to (\mathtt{comm}, r)$ which takes as input a message $m \in \mathcal{M}$ and outputs a commitment $\mathtt{comm} \in \mathcal{C}$ and masking randomness $r \in \mathcal{N}$. Slightly overloading notation, we write $\mathtt{comm} \leftarrow \mathsf{Commit}(m; r)$ to denote resulting commitment for a predetermined choice of $r$.

3. $\mathsf{Open}(\mathtt{comm}, m, r; \mathsf{pp}) \to \{0, 1\}$ which takes as input a commitment $\mathtt{comm}$, message $m$ and masking randomness $r$ and outputs a bit. If the output is 1, the message $m$ is said to be a valid opening of the commitment $\mathtt{comm}$.

For the purpose of this thesis, a commitment scheme is said to be secure if it is computationally binding and {*perfectly, statistically, computationally*} hiding. We define these properties below:

**Definition 1** (Computationally Binding)**.** A cryptographic commitment scheme $\Pi = (\mathsf{SetUp}, \mathsf{Commit}, \mathsf{Open})$ is said to computationally binding if an adversary cannot produce two distinct messages $m \neq m'$ that open to the same commitment $\mathtt{comm}$ with non-negligible probability. Formally, for all PPT adversaries $\mathcal{A}$, and all $\lambda \in \mathbb{N}$, the following holds:

$$\Pr\left[\begin{array}{c} \mathsf{Open}(\mathtt{comm}, m, r; \mathsf{pp}) = \\ \mathsf{Open}(\mathtt{comm}, m', r'; \mathsf{pp}) = 1 \wedge m \neq m' \end{array} \middle| \begin{array}{c} (\mathtt{comm}, m, m', r, r') \leftarrow \mathcal{A}(\mathsf{pp}), \\ \mathsf{pp} \leftarrow \mathsf{SetUp}(1^\lambda) \end{array}\right] \leq \mathrm{negl}(\lambda)$$

where the probability is taken over the randomness of $\mathcal{A}$ and $\mathsf{SetUp}$.

**Definition 2** (Hiding)**.** A commitment scheme $\Pi = (\mathsf{SetUp}, \mathsf{Commit}, \mathsf{Open})$ is said to be {*perfectly, statistically, computationally*} hiding if, for all $\lambda \in \mathbb{N}$, public parameters $\mathsf{pp} \leftarrow \mathsf{SetUp}(1^\lambda)$, and distinct $m, m' \in \mathcal{M}$, the following probability distributions

$$\{\mathsf{Comm}(m, r; \mathsf{pp}) \mid r \leftarrow\!\!\$\, \mathcal{N}\}, \text{ and } \{\mathsf{Comm}(m', r'; \mathsf{pp}) \mid r' \leftarrow\!\!\$\, \mathcal{N}\}$$

are {*perfectly, statistically, computationally*} indistinguishable.

Pedersen commitments, first introduced in [Ped92], are relevant to Chapter 3 and Section 6.2. We describe the Pedersen commitment scheme below in Figure 2.1 generically for groups of prime order. We note that Pedersen commitments are not computationally binding in the presence of a quantum adversary (who can perform Shor's algorithm [Sho97] to compute discrete logarithms of a group), and hence they are not post-quantum secure.

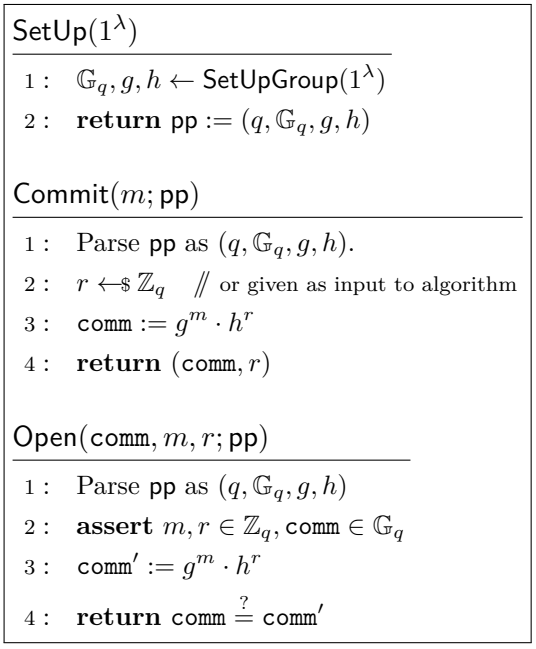



Figure 2.1: The Pedersen commitment scheme [Ped92] defined generically for groups of prime order. Let SetUpGroup be a black-box procedure which takes as input $1^\lambda$ and outputs a group $\mathbb{G}_q$ of prime order $q$ (with group operation $\cdot$) where computing discrete logarithms classically is computationally infeasible; as well as two generators $g, h$ of $\mathbb{G}_q$ such that the discrete logarithm of $g$ with respect to $h$ is unknown.

**Theorem 1** (Folklore)**.** *The Pedersen commitment scheme described in Figure 2.1 is computationally binding (assuming the hardness of computing discrete logarithms) and perfectly hiding.*

Pedersen commitments also satisfy the *additive homomorphic* property, meaning that

$$\texttt{comm}(m_1; r_1) \cdot \texttt{comm}(m_2; r_2) = \texttt{comm}(m_1 + m_2; r_1 + r_2),$$

for all $m_1, m_2 \in \mathcal{M}$ and $r_1, r_2 \in \mathcal{N}$, where multiplication on the left hand side is the underlying group operation of $\mathcal{C}(= \mathbb{G}_q)$ (in the case of elliptic curve groups, this is point addition; and in the case of the ring of integers modulo $q$, this is modular multiplication), and the addition is performed in the message space $\mathcal{M}(= \mathbb{Z}_q)$ as an additive group. Note that in this section we refer to this group operation multiplicatively, and we continue to do so in Chapter 3, but we will refer to the group operation additively in Sections 6.2 and 6.3, as the focus is on elliptic curve operations, where the multiplicative operator may cause confusion.

## 2.2 Zero-Knowledge Proofs of Knowledge

Let $\mathcal{R} : X \times W$ be a relation with input set $X$ and witness set $W$ defining the NP-language $\mathcal{L} = \{x \in X : \exists w \in W \text{ s.t. } (x, w) \in \mathcal{R}\}$. Zero-knowledge proofs are protocols between two parties, where a prover $P$ tries to convince a verifier $V$, given some $x \in X$, that $x \in \mathcal{L}$, i.e. there exists (or it knows) a witness $w \in W$ such that $(x, w) \in \mathcal{R}$ [GMR89].

If this witness does indeed exist and the proof is generated correctly, then the verifier should accept the proof. On the other hand, if no such witness exists, the prover should not be able to convince a verifier that it does. These fundamental security conditions are called *completeness* and *soundness* of the zero-knowledge proof, respectively. *Knowledge soundness*, a stronger notion of soundness first introduced in [BG93], does not only convince a verifier that the witness exists, but also that the prover knows it. Formally, this is shown by defining an algorithm called the extractor, which has oracle access to the prover and tries to derive the witness from this interaction. Finally, the notion of *zero-knowledge* guarantees that the verifier does not learn any information about the witness, except from what it can derive from publicly available data. This last property is proven by showing the existence of a simulator, which can generate protocol transcripts that are indistinguishable from those between an honest prover and verifier on the same instance, even without knowing the witness.

Zero-knowledge proofs may be interactive protocols, but can generally be made non-interactive in the Random Oracle Model [BR93] via the Fiat-Shamir transform [FS87], or its many variants [BCS16, Unr17]. We are primarily interested in non-interactive zero-knowledge proofs *of knowledge* in this work and will, on occasion, abbreviate them as either NIZK or NIZKPoK. We further write $\mathsf{H}$ or $O$ for an invocation of a random oracle. We outline general definitions of the security properties of a NIZKPoK below (in the Random Oracle Model). We refer the reader to [GOP+25, AFK23] for a more comprehensive treatment of non-interactive proofs obtained via the Fiat-Shamir transform.

**Definition 3** (Completeness)**.** For every $(x, w) \in \mathcal{R}$, honest prover $P$ and honest verifier $V$, it holds that

$$\Pr[V^O(x, \pi) = 1 \mid \pi \leftarrow P^O(x, w)] = 1$$

**Definition 4** (Soundness)**.** For any $x \notin \mathcal{L}$, (potentially malicious) PPT prover $P$ and honest verifier $V$, it holds that

$$\Pr[V^{\mathsf{H}}(x, \pi) = 1 \mid \pi \leftarrow P^{\mathsf{H}}(x)] \leq \mathsf{negl}(\lambda)$$

**Definition 5** (Knowledge Soundness)**.** We say that a protocol is knowledge sound, with knowledge error $\kappa$, if there exists a $\mathsf{PPT}$ extractor $E$ and positive polynomial $q$ such that, for every $x$, $\mathsf{PPT}$ $\tilde{P}$, $\lambda \in \mathbb{N}$,

$$q(|x|) \cdot \Pr[(x, w) \in \mathcal{R} \mid w \leftarrow E^{\tilde{P}}(x, 1^\lambda)] \geq \Pr[V^H(x, \pi) = 1 \mid \pi \leftarrow \tilde{P}^H] - \kappa(|x|, \lambda).$$

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 $\pi$, at which point $\tilde{P}$ goes to the start of its computation with the same randomness and auxiliary input.

**Definition 6** (Zero Knowledge)**.** A non-interactive protocol $(P, V)$ is $\{$*computational, statistical, perfect*$\}$ zero-knowledge (with security parameter $\lambda$) in the random oracle model, if there exists a PPT simulator $\mathcal{S}$, such that for every $(x, w) \in \mathcal{R}$, the following distributions:

$$\{\pi \leftarrow \mathcal{S}^H(x)\}, \text{ and } \{\pi \leftarrow P^H(x, w)\},$$

are $\{$*computationally, statistically, perfectly*$\}$ indistinguishable, where the distributions are taken over the uniformly random instantiation of $H$ and the randomness of $P$. Note that $H$ is an explicitly programmable random oracle, where $\mathcal{S}$ may program responses to specific inputs of $\mathcal{S}$'s choice.

A common approach to constructing signature schemes is by applying the Fiat-Shamir transform to an interactive proof system, and adding the message to the query to the random oracle. In order to guarantee such a protocol yields a signature scheme that is *fully existentially unforgeable under chosen message attack* (EUF-CMA), the underlying NIZKPoK must satisfy a stronger notion of *adaptive* knowledge soundness, also called *simulation extractability*. While Definition 5 is defined for a static adversary $\tilde{P}$ for a fixed instance $x$, adaptive knowledge soundness requires that knowledge soundness holds even when the prover is allowed to choose the instance $x$ on the fly based on the random oracle queries it has made so far. For a formal definition, see [AFK23, Defn. 10] [GOP+25, Defn 2.8].

In this work, we consider NIZKPoKs constructed via two approaches, which are common in the literature. Those constructed from the Fiat-Shamir transformation of $\Sigma$-protocols, and those constructed from non-interactive transformations of interactive oracle proofs (IOPs). The key differentiating factor is that the latter results in a non-interactive protocol which is also *succinct*: the proof size scales polylogarithmically in the size of the witness. We shall briefly discuss other differentiating features of these two approaches below, and outline the definitions of the underlying protocols in their respective sections.

**zk-SNARKs versus NIZKPoKs from Sigma protocols** While $\Sigma$-protocols provide the gold-standard approach for constructing cryptographic protocols which naturally arise from interactive proofs (such as verifiable random functions, identification protocols and digital signatures), they do not always offer a practical level of efficiency, particularly in the case of the post-quantum setting of isogenies. Indeed, in this work we will use succinct non-interactive arguments of knowledge (zk-SNARKs) to prove knowledge of isogeny paths. It is worth noting that the underlying IOP is often multiple round (even

variable round in the length of the input), and the simulation extractability of the resulting non-interactive protocol is not always well understood, but recent frameworks have been proposed to address this [KPT23, FFK+23].

### 2.2.1 Sigma protocols

For a comprehensive introduction to Σ-protocols, we refer the reader to [Dam10]. We introduce the basic definitions and security properties of Sigma protocols here. A Σ-*protocol* (or *Sigma protocol*) for a relation $\mathcal{R}$ is a 3-round public coin interactive proof performed between prover $P$ on input $(x, w)$ and a verifier $V$ on input $x$:

**Commitment:** A prover sends a message $\mathsf{comm}$ to the verifier.

**Challenge:** On receiving $\mathsf{comm}$ sends a uniformly sampled $\mathsf{chall} \leftarrow\!\!\$\, \mathcal{C}$ from a challenge space $\mathcal{C}$.

**Response:** on receiving $\mathsf{chall}$, sends a response $\mathsf{resp}$.

**Verification:** After the 3 rounds of interaction, on input $(x, \mathsf{comm}, \mathsf{chall}, \mathsf{resp})$ the verifier outputs either 1 (accept) or 0 (reject).

A Σ-protocol must also satisfy completeness, special soundness, and honest verifier zero-knowledge. In the context of sigma protocols, completeness holds if a verifier always accepts an honest prover. We include two variants of the latter property, which are called *weak* and *strong* honest verifier zero-knowledge. Looking ahead, the distinction will arise where knowledge of the witness can allow an adversary to distinguish real and simulated transcripts, as in Section 6.4. In some cases it is required to use a sigma protocol which satisfies the stronger variant, such as in the construction of fully-anonymous ring signatures [AOS02]. The only difference is that in the strong variant, the witness to the instance is included in the distributions. We note that the properties are equivalent if the sigma protocol is statistically or perfectly zero-knowledge, and differ only in the computational setting. Hence, where a protocol is perfectly or statistically zero-knowledge, we may omit this distinction (such as in Section 6.3).

**Definition 7** (Strong Honest Verifier Zero-Knowledge (HVZK))**.** A Σ-protocol is said to be *strong* {*perfectly, statistically, computationally*} zero-knowledge if there exists a PPT simulator $\mathcal{S}$ who makes at most polynomially many queries to the random oracle $O$, such that for every $(x, w) \in \mathcal{R}$ and $\lambda \in \mathbb{N}$, the following distributions are {*perfectly, statistically, computationally*} indistinguishable:

$$\{(x, w, \langle \mathcal{P}^O(x, w), \mathcal{V}^O(x) \rangle)\}, \text{ and } \{(x, w, \mathcal{S}^O(x))\}.$$

where $\langle \mathcal{P}^O(x,w), \mathcal{V}^O(x)\rangle$ is the transcript of a protocol execution between an honest prover and verifier, and the distributions are taken over the random coins of $\mathcal{P}$, $\mathcal{V}$ and $\mathcal{S}$.

**Definition 8** (Weak Honest Verifier Zero-Knowledge (HVZK))**.** A $\Sigma$-protocol is said to be *weak* {*perfectly, statistically, computationally*} zero-knowledge if there exists a PPT simulator $\mathcal{S}$ who makes at most polynomially many queries to the random oracle $O$, such that for every $(x,w) \in \mathcal{R}$ and $\lambda \in \mathbb{N}$, the following distributions are {*perfectly, statistically, computationally*} indistinguishable:

$$\{(x, \langle \mathcal{P}^O(x,w), \mathcal{V}^O(x)\rangle)\}, \text{ and } \{(x, \mathcal{S}^O(x))\}.$$

where $\langle \mathcal{P}^O(x,w), \mathcal{V}^O(x)\rangle$ is the transcript of a protocol execution between an honest prover and verifier, and the distributions are taken over the random coins of $\mathcal{P}$, $\mathcal{V}$ and $\mathcal{S}$.

*Remark* 1. We note that in the interactive setting, we do not explicitly require a programmable oracle. In this setting, the simulator generally functions by sampling a uniform challenge first (corresponding to the verifier's choice), and computes a commitment and response phase such that the transcript is accepting and identically distributed to real protocol executions.

The greatest advantage of $\Sigma$-protocols is their notion of knowledge soundness, called *2-special soundness*, or *t-special soundness* which generalises the former. This property is convenient for proving knowledge soundness of the resulting protocol.

**Definition 9** ($t$-special soundness)**.** There exists a PPT algorithm $E$ called the extractor, which given instance $x$, and $t$ valid distinct transcripts

$$[(\mathtt{comm}_i, \mathtt{chall}_i, \mathtt{resp}_i)_{i\in[t]}]$$

where $\mathtt{comm}_i = \mathtt{comm}_j$ (with a common first message), $\mathtt{chall}_i \neq \mathtt{chall}_j$ for all $1 \leq i < j \leq t$, $E$ outputs $w$ such that $(x,w) \in \mathcal{R}$.

When the Fiat-Shamir [FS87] transform is applied to a Sigma protocol which is complete, $t$-special sound, and honest verifier zero-knowledge, the resulting protocol is a NIZKPoK with zero-knowledge, and knowledge error $\frac{t-1}{c}$ where $c$ is the size of the verifier's challenge space.

### 2.2.2 zk-SNARKs

In the (explicitly programmable) random oracle model, a *zero-knowledge non-interactive succinct argument*[1] *of knowledge* (zk-SNARK) for a relation $\mathcal{R} = \{(x,w)\}$ is a tuple

[1]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.

$(P, V)$ where $P, V$ are PPT algorithms with access to a random oracle $O$ which satisfy the properties of Definitions 3 to 6, and are also *succinct*, as defined below.

**Definition 10** (Succinctness)**.** A proof system $(P, V)$ for a relation $\mathcal{R}$ is *succinct*, if, for any $(x, w) \in \mathcal{R}$ and corresponding proof $\pi \leftarrow P^O(x, w)$, $\pi$ grows polylogarithmically in $w$. In particular, $|\pi| = \mathsf{poly}(\lambda, |x|, \log(|w|))$.

The zk-SNARKs we consider in this work are *transparent*[2], in the random oracle model, and constructed from *Interactive Oracle Proofs* (IOPs) via the BCS transform [BCS16].

**Interactive Oracle Proofs** An *interactive oracle protocol* between two PPT algorithms $A$ and $B$ over $k$ rounds is a protocol where at the $i$-th 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 subsequent 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 soundness $s$ satisfies 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, zk-SNARKs can be obtained from IOPs. Intuitively, succinct proofs are achievable when the prover sends random access oracles, which are then realised using efficient polynomial commitment schemes (PCS), which allow a prover to commit to a polynomial and later open it at a point of the verifier's choice (without necessarily revealing the polynomial).

**Theorem 2** (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*

[2]Information theoretic, not requiring any trusted setup or computational assumptions.

*zero-knowledge, the resulting protocol is statistically zero-knowledge under the explicitly programmable random oracle model.*[3]

*Proof.* See [BCS16, Sec. 6] □

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

### 2.2.3 Rank-1 Constraint Systems

SNARKs are designed to work for generic relations, which can encode arbitrary computations. A common relation used to encode general arithmetic circuits is *Rank-1 Constraint Systems* (R1CS). First, we define the coordinate-wise Hadamard product. Given two vectors in $\mathbb{F}_q^n$:

$$\begin{pmatrix} a_1 \\ \dots \\ a_n \end{pmatrix} \circ \begin{pmatrix} b_1 \\ \dots \\ b_n \end{pmatrix} = \begin{pmatrix} a_1 b_1 \\ \dots \\ a_n b_n \end{pmatrix}$$

Now, the R1CS relation is defined as follows:

$$\mathcal{R}_{\text{R1CS}} = \{(A, B, C, \mathbf{v}, q), (\mathbf{w}) \mid A\mathbf{z} \circ B\mathbf{z} = C\mathbf{z}, \mathbf{z} = (1 \ \mathbf{v} \ \mathbf{w})\}.$$

R1CS is parameterised by an underlying finite field $\mathbb{F}_q$, and consists of instance-witness pairs $((A, B, C, \mathbf{v}), \mathbf{w})$ where $A, B, C \in \mathbb{F}_q^{m \times (n+1)}$ and $\mathbf{v}$,$\mathbf{w}$ are vectors over $\mathbb{F}_q$ such that

$$Az \circ Bz = Cz$$

for $z := (1 \ \mathbf{v} \ \mathbf{w}) \in \mathbb{F}_q^{n+1}$. Conceptually, $A, B, C$ encode constraints on variables $\mathbf{v}, \mathbf{w}$; where $\mathbf{v}$ contains (public) auxiliary input, and $\mathbf{w}$ contains both secret input and intermediate variables in a computation. R1CS may encode arithmetic circuit satisfiability. In loose terms, in R1CS, each row can encode: *linear expression* $\times$ *linear expression* $=$ *linear expression*. In Chapter 4, we encode several isogeny relations in compact R1CS instances, which when combined with a suitable zk-SNARK, can be used in an isogeny-based protocol, such as our Verifiable Random Function (VRF) construction in Section 5.3. In Section 3.5, we describe a straightforward way to prove $\mathcal{R}_{\text{R1CS}}$ using $\Sigma$-protocols in the discrete logarithm setting.

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

**Example** Suppose we wish to encode the following polynomial equation in $\mathbb{F}_q[a, b, c, d]$ into an R1CS instance (supposing all variables are secret):

$$8a^3 + 6ab + c + 5d = 50.$$

In order to make this equation amenable to R1CS, we first rearrange it into a system of equations which are in the form: *linear expression* $\times$ *linear expression* $=$ *linear expression*. To do this, we add an intermediate variable $e$ to the equation, and rewrite it as:

$$a^2 = e$$
$$2a \cdot (4e + 3b) = 50 - c - 5d$$

This will yield an R1CS instance with 5 variables and 2 constraints. More precisely, our R1CS instance-witness pair is $(A, B, C, v, q), w$ where $z = (1\ v\ w) = (1, a, b, c, d, e)$, and $A, B, C$ are the following matrices:

$$A = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 \end{pmatrix}$$
$$B = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 & 4 \end{pmatrix}$$
$$C = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 1 \\ 50 & 0 & 0 & -1 & -5 & 0 \end{pmatrix}$$

## 2.3 Isogeny-Based Cryptography

In this section we introduce the necessary background on elliptic curves and isogenies for cryptographic applications relevant to this thesis. We refer the reader to [Sil86] for a comprehensive overview of the underlying mathematical background.

### 2.3.1 Elliptic curves and isogenies

**Elliptic Curves** An elliptic curve is a non-singular projective curve of genus one. An elliptic curve $E$ may represented by the general equation:

$$E : y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6$$

where $E$ is said to be defined over a field $\mathbb{F}$ if its coefficients $a_1, \ldots a_6$ are in $\mathbb{F}$. In our work related to radical isogenies (see [CDV20]), we consider elliptic curves of the form

$$E : y^2 = x^3 + ax^2 + cx \tag{2.1}$$

where $c \neq 0$ and $a^2 - 4c \neq 0$.

**The $j$-invariant** The *$j$-invariant* of an elliptic curve is an element of $\mathbb{F}$ given as output of a function $j$ which takes as input the coefficients of curve defined over a field $\mathbb{F}$. It is well defined for all curve models and is isomorphism invariant. That is, $j(E) = j(E')$ if and only if $E$ and $E'$ are isomorphic over $\overline{\mathbb{F}}$. As an example, given a curves $E$ represented in the model of Equation (2.1):

$$j(E) := 256 \cdot \frac{(a^2 - 3c)^3}{c^2(a^2 - 4c)}.$$

**The Elliptic Curve Group of Points** The $\mathbb{F}$-rational points of a curve $E(\mathbb{F})$ are the set of affine solutions to the curve equation over $\mathbb{F}$, plus an additional point at infinity. It is well known that $E(\mathbb{F})$ forms an additive abelian group under a defined addition law $+ : E \times E \to E$, with the identity element given by the point at infinity, hence we refer to this point as 0. The $N$-torsion of $E(\mathbb{F})$, denoted as $E[N]$ (or $E(\mathbb{F})[N]$ more precisely), is the set of points of order $N$:

$$E[N] = \{P \in E(\mathbb{F}) \mid [N]P = 0\}.$$

When $\gcd(N, \text{char } \mathbb{F}) = 1$, over the algebraic closure, the group $E[N]$ is isomorphic to $\mathbb{Z}/N\mathbb{Z} \times \mathbb{Z}/N\mathbb{Z}$, and is thus always 2-generated by a basis of points $P_N, Q_N$.

**Isogenies** An *isogeny* is a surjective morphism between elliptic curves of finite kernel, which acts as a group homomorphism between their group of points over a field. An isogeny $\phi$ is *separable* if it is determined solely by its kernel, denoted $\ker \phi$. The degree of a separable isogeny is given by its degree as a rational map and the size of its kernel. An isogeny is *cyclic* if its kernel is a cyclic group. Given a point $P \in E$ of order $N$, we may write the codomain of an associated isogeny, with kernel generated by $P$, as $E/\langle P \rangle$. For every isogeny $\phi : E \to E'$, there exists a unique dual isogeny $\hat{\phi} : E' \to E$ such that $\hat{\phi} \circ \phi = [\deg \phi]$, where $[m]$ denotes the multiplication-by-$m$ map of a curve. Given there exists an isogeny $\phi : E \to E'$ of degree $N$, we say that $E$ and $E'$ are $N$-isogenous, and that $\phi$ is an $N$-isogeny. In this work, we assume all isogenies are separable and cyclic (unless otherwise stated).

**Endomorphism Rings and Supersingular Curves** Let $E$ be an elliptic curve over a field $\mathbb{F}$. Then $\text{End}(E)$ is the endomorphism ring of $E$, i.e., the set of isogenies from $E$ to itself. Over a finite field $\mathbb{F}_{p^2}$, the endomorphism ring of elliptic curves either form (i) an imaginary quadratic field, in which case we say the curve is *ordinary*, or (ii) a maximal order $\mathcal{O}$ in a quaternion algebra, in which we say the curve is *supersingular*. In this work, and in the wider context of cryptographic applications, we are interested in the latter case. For a curve $E$ defined over a field of characteristic $p$, $E[p^r]$ is trivial for

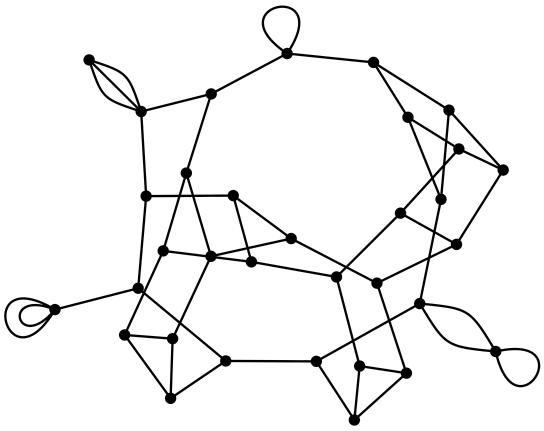


Figure 2.2: The supersingular 2-isogeny graph over $\mathbb{F}_{383^2}$.

all $r$ if and only if $E$ is supersingular. Moreover, if $E$ is supersingular then $j(E) \in \mathbb{F}_{p^2}$, so we are only typically interested in working over at most the quadratic extension.

**The supersingular $\ell$-isogeny graph** The *supersingular $\ell$-isogeny graph over $\mathbb{F}_{p^2}$*, denoted as $G_\ell(p)$, is a graph constructed by taking the vertices to be the set of supersingular elliptic curves over $\mathbb{F}_{p^2}$ up to isomorphism (often labelled by their $j$-invariant), and the edges to be all $\ell$-isogenies between curves. We may view such a graph as undirected by identifying every isogeny with its dual[4]. It is a well known fact that the graph is connected and $\ell + 1$-regular for all primes $\ell$. First introduced by Pizer [Piz90], these graphs are *Ramanujan*, which means that the distribution of random walks taken on the graph quickly approaches the uniform distribution on the vertex set. This graph construction is helpful as it allows us to view $\ell^k$-isogenies as length $k$ walks in $G_\ell(p)$. Given an isogeny $\phi : E_0 \to E_k$ of degree $\ell^k$ with kernel $\langle G \rangle$, we may decompose it into a chain of $\ell$-isogenies where:

$$\phi = \phi_k \circ \phi_{k-1} \circ \cdots \circ \phi_1$$

where $\phi_i : E_{i-1} \to E_i$ for $i = 1, \ldots, k$ are $\ell$-isogenies where $\ker \phi_1 = \langle [\ell^{k-1}]G \rangle$ and $\ker \phi_{i+1} = \langle [\ell^{k-i-1}]\phi_i \circ \cdots \circ \phi_1(G) \rangle$ for $i \geq 1$. The converse also holds, and hence an isogeny $\phi : E_0 \to E_k$ of degree $2^k$ is cyclic if and only if $\phi$'s decomposition into 2-isogenies with curves as a walk on the supersingular isogeny graph is *non-backtracking* (see [CLG09, Prop. 1]). We define non-backtracking to mean that every outgoing edge of the walk is never the dual of the in-going edge, i.e. a non-backtracking walk written as a composition of $\ell$-isogenies: $\phi_1 \circ \phi_2 \circ \cdots \circ \phi_k$ does not have $\phi_{i+1} = \hat{\phi}_i$ for any $i$.

[4]The only obstructions to writing the graph as undirected occur at $j = 0, 1728$.

**Modular Polynomials** 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 $\ell$-isogenous $j$-invariants of elliptic curves over $\mathbb{F}_{p^2}$. This allows us to efficiently determine if two elliptic curves are $\ell$-isogenous over a given field. In particular, two $j$-invariants of elliptic curves $j_1, j_2$ are adjacent in $G_\ell(p)$ if and only if $\Phi_\ell(j_1, j_2) = 0 \mod p$. As an example, for $\ell = 2$, we have the modular polynomial

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

**Radical Isogenies** The work of [CDV20] introduces techniques which allow for a more efficient way to compute non-backtracking $\ell$-isogeny chains for arbitrary $\ell$. More precisely, given an elliptic curve $E$ and a prescribed point $P$ on $E$ of order $\ell$, the paper describes an approach to construct a point $P'$ on $E' = E/\langle P\rangle$ such that the composition of isogenies

$$E \to E/\langle P\rangle \to E'/\langle P'\rangle$$

is a cyclic (or non-backtracking) isogeny of degree $\ell^2$. This technique can then be extended in sequence to compute arbitrary length $\ell$-power cyclic isogenies. The formulae take advantage of the structure present in the elliptic curve model, and rely on computing an $\ell$-th root, which also determines the choice of the outgoing $\ell$-isogeny at each step. We include the formulae in the case of $\ell = 2$ later in Section 4.2, but refer the reader to [CDV20] for the general case and more details on how the formulae were derived.

### 2.3.2 Isogeny-based Cryptography

**CGL Hash Function** The first isogeny-based cryptographic primitive was the CGL hash function, introduced by Charles, Goren and Lauter in [CLG09]. The function, given a public supersingular starting curve $E$ (typically represented by $j$-invariant) over $\mathbb{F}_{p^2}$, takes a non-backtracking walk in the supersingular $\ell$-isogeny graph, dictated by some input $k$. This can be done by parsing the input string in base $\ell$, e.g. $k = k_0k_1 \dots k_n$ with $k_i \in \{0, \dots, \ell - 1\}$ and ordering the outgoing (non-backtracking) edges from every vertex $E_i$ in some canonical way, then choosing the path that corresponds to the digit $k_i$. After $n$ steps, one arrives at the output curve $E_n$ (again, typically represented by its $j$-invariant). The CGL hash function is provably preimage resistant given the hardness of one of the most general isogeny-based computational assumptions below, which also underpins the security of many isogeny-based protocols.

**Problem 1** ($\ell^n$-isogeny path problem)**.** *Given two supersingular elliptic curves $E, E'$ over $\mathbb{F}_{p^2}$, if it exists, find a cyclic isogeny $\phi : E \to E'$ of degree $\deg \phi = \ell^n$.*

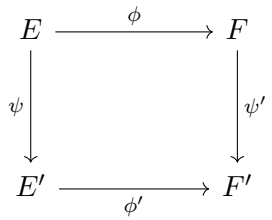



Figure 2.3: An SIDH square, where $\phi$ and $\phi'$ are $N$-isogenies, and $\psi$ and $\psi'$ are $N'$-isogenies, such that $\ker \phi' = \psi(\ker \phi)$ and $\ker \psi' = \phi(\ker \psi)$.

The CGL function is collision resistant provided either the endomorphism ring of the starting curve is unknown, or the input length is constrained enough to prevent the existence of cycles of length less than twice the input length. A convenient way of instantiating the CGL hash function is by using the supersingular 2-isogeny graph, which is 3-regular. Thus, by preventing backtracking, the direction of each step of the walk is determined by a single bit of the input.

**Computing Isogenies in the SIDH Setting** Over $\mathbb{F}_{p^2}$, given that $N \mid p \pm 1$, we have that the $N$-torsion group $E[N] \cong (\mathbb{Z}/N\mathbb{Z})^2$, and can be generated by a basis of independent points $\langle P_N, Q_N \rangle$. This yields a convenient way of representing isogenies of degree $N$ by computing their kernels as linear combinations of the basis generators. Given a kernel $G = \langle [a]P_N + [b]Q_N \rangle$ for $a, b \in \mathbb{Z}_N$, an isogeny $\phi : E \to E/G$ can be computed using Vélu's formulas [Vél71] in $O(N)$ time[5]. The SIDH protocol[JD11] was a key exchange protocol that exploited the following structure of isogenies: suppose $N, N'$ are coprime integers such that $NN' \mid p \pm 1$, and let $\psi : E \to E'$ be an $N'$-isogeny. Then $\psi$ has $\mathbb{F}_{p^2}$-rational kernel and acts as an invertible linear transformation on the $N$-torsion group $E[N]$ (i.e. $\psi([a]P + [b]Q) = [a]\psi(P) + [b]\psi(Q)$ for $P, Q \in E[N]$).

**Definition 11** (SIDH Square)**.** An SIDH square (see Fig. 2.3) is a tuple:

$$(p, E, F, E', F', \phi, \psi, \phi', \psi')$$

which satisfy the following conditions:

- $E, F, E', F'$ are supersingular elliptic curves over $\mathbb{F}_{p^2}$.
- $\phi : E \to F$, $\phi' : E' \to F'$ are $N$-isogenies and $\psi : E \to E'$, $\psi' : F \to F'$ are $N'$-isogenies such that $gcd(N, N') = 1$ and $NN' | p \pm 1$.
- $\ker \phi' = \psi(\ker \phi)$ and $\ker \psi' = \phi(\ker \psi)$, hence $\psi' \circ \phi = \phi' \circ \psi$.

[5]Additionally, one can guarantee that distinct inputs map to distinct isogenies by mapping an input $m \in \mathbb{Z}_N$ to isogeny with kernel $P_N + [m]Q_N$, as was the case in SIDH.

**SIDH Key Exchange** In the SIDH protocol, two participants interact in a key exchange where the kernels of their isogenies (corresponding to the horizontal and vertical isogenies in Figure 2.3 respectively) are kept secret such that $\ker \phi = \langle P_N + [s]Q_N \rangle$ and $\ker \psi = \langle P_{N'} + [s']Q_{N'} \rangle$. In order to compute the push-through isogenies $\phi'$ (resp. $\psi'$), each party would also publish the image of the coprime torsion basis, $\psi(P_N), \psi(Q_N)$ (resp. $\phi(P_{N'}), \phi(Q_{N'})$). The SIDH protocol was rendered broken by attacks [MMP+23, CD23, Rob23], which exploited the fact that an attacker, who wished to learn the kernel of the vertical isogeny $\psi$ could use the action of $\psi$ on $(P_N, Q_N)$ to recover the secret $s'$. It is worth noting protocols using SIDH square structure are not necessarily susceptible to the polynomial time SIDH attack, that does not apply when the protocol does not leak the action of an isogeny on a coprime torsion basis, which is the case in a number of follow up works [FMP23, Bas24a, Bas24b, BMP23].

Works on proving knowledge of isogenies [JD11, DDGZ22, GPS17, BCC+23] exploit the structure of SIDH squares in order to construct sigma protocols serving as zero-knowledge proofs of knowledge. These can be compiled via the Fiat-Shamir transform to obtain a number of cryptographic protocols, such as digital signatures [GPS17], Oblivious Pseudorandom Functions (OPRFs) [Bas24b, BKW20] and Verifiable Delay Functions (VDFs) [DMPS19]. Of relevance to our work, is the the proof of knowledge of an isogeny presented in [DDGZ22], which relies on the introduction of a decisional assumption. In particular, this assumption does not leak torsion information, and is therefore not susceptible to the SIDH attacks from [MMP+23, CD23, Rob23].

**Problem 2** (Decisional Supersingular Product (DSSP) Problem)**.** *Given an isogeny $\phi : E \to F$ of degree $\ell_1^{n_1}$, construct a PPT distinguisher $\mathcal{A}$ with non-negligible advantage for the following two distributions:*

- $\mathcal{D}_0 = \{(E', F', \phi')\}$ *such that $\phi' : E' \to F'$ is a $\ell_1^{n_1}$-isogeny, and there exists an isogeny $\psi : E \to E'$ of degree $\ell_2^{n_2}$ such that $\ker \phi' = \psi(\ker \phi)$. The distribution is taken over the uniform sampling of $\psi$.*
- $\mathcal{D}_1 = \{(E', F', \phi')\}$ *such that $\phi' : E' \to F'$ is a $\ell_1^{n_1}$-isogeny, and $E'$ has the same order as $E$. The distribution is taken over the uniform sampling of the curve $E'$, and isogeny $\phi'$.*

We say that isogenies $\phi, \phi'$ are *parallel* if they can exist as parallel sides of an SIDH square. In particular, when there exists coprime degree isogenies $\psi : E \to E', \psi' : F \to F'$ such that the conditions of Definition 11 are satisfied; namely that $\ker \phi' = \psi(\ker \phi)$ and $\ker \psi' = \phi(\ker \psi)$. Problem 2 asks a distinguisher which is given two isogenies to determine if they are parallel.

Let $\mu_N = \{x \in \mathbb{F} \ : \ x^N = 1\}$ be the set of $N$-th roots of unity. The *Weil pairing* $e_N : E[N] \times E[N] \to \mu_N$ is an alternating, bilinear map on the points of order $N$ of an elliptic curve over $\mathbb{F}$. We also have the following.

**Proposition 1** ([Sil86, Prop. III.8.2])**.** *Let $N, N'$ be coprime integers and $\phi : E \to E'$ be an $N'$-isogeny over $\mathbb{F}$. Then for all $P, Q \in E[N]$, it holds that $e_N(\phi(P), \phi(Q)) = e_N(P, Q)^{N'}$.*

# Chapter 3

# Proving Arithmetic Statements on Pedersen Commitments

We begin the novel contributions of the thesis with a "warm-up" chapter, which collects and proves some folklore results around proving general arithmetic statements on commitments using Σ-protocols.

The work in this chapter relies on Pedersen commitments (see their definition in Section 2.1.1), which renders all techniques secure strictly only in the classical model (in the absence of quantum computational attacks), but the results trivially generalise to any perfectly hiding, computationally binding, additively homomorphic commitment schemes. Many classical proof of knowledge constructions rely on Σ-protocols proving arithmetic relations between different Pedersen Commitments. The area was largely pioneered by Schnorr [Sch91], Chaum-Pedersen [CP93], Fujisaki-Okamoto [FO97] and Camenisch et. al. [CM99, CS97] In this section, we mostly introduce the proofs of [WTs+18, App. A] due to their efficiency, and prove security of a folklore construction in Fig. 3.4. We also introduce an approach to proving R1CS statements using Pedersen commitments in Section 3.5, using only the proofs we introduce in this section.

**Statement of Authorship Contribution (Chapter 3)** *The following section is adapted from my contributions to [CLR24], which collects various results from other works on proofs of Pedersen commitments – namely [Sch91, WTs+18]. All of the content included in this chapter which was not in my own words has either been rewritten or expanded upon, and I am solely responsible for the intellectual contributions of the content present in this chapter.*

## 3.1 Proving Knowledge of an Opening to a Commitment

Given a commitment $C$ to a message $x$ with randomness $r$, a prover may convince a verifier that they know a valid opening to the commitment by engaging in the protocol described in Fig. 3.1.

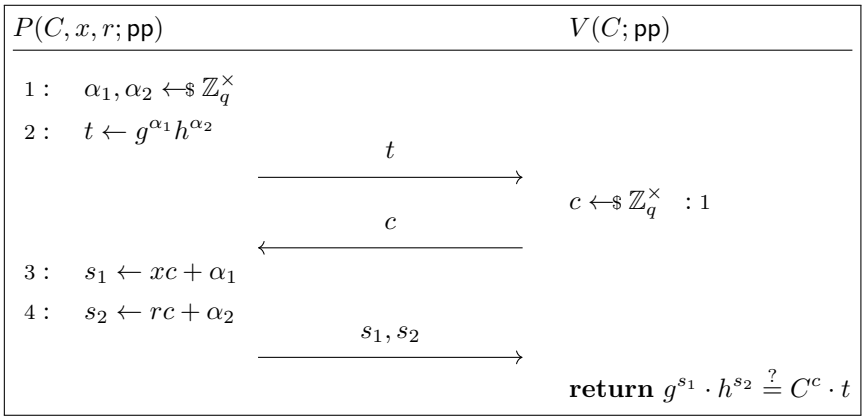



Figure 3.1: Sigma protocol for proving knowledge of an opening of a Pedersen commitment. We denote a parallel composition of this proof by calling OpenProof with the prover's input.

**Theorem 3.** *Fig. 3.1 is a $\Sigma$-protocol for the relation:*

$$\mathcal{R} = \{((C, g, h, q), (x, r)) \mid C = g^x h^r\}.$$

*Proof.* Follows from [Sch91]. □

## 3.2 Proving Commitments Open to the Same Message

Given commitments $(C_1, C_2)$ to the same message $x$ under different randomness $C_1 = \mathsf{Commit}(x, r_1)$ and $C_2 = \mathsf{Commit}(x, r_2)$, a prover may engage in Figure 3.2. Observe that if $C_1$ and $C_2$ do not open to the same message, there is still a satisfying witness to the relation in Theorem 4. Taking advantage of it, however, requires knowledge of a discrete logarithm of $C_1 C_2^{-1}$ to the base $h$.

**Theorem 4** (Folklore)**.** *Fig. 3.2 is a $\Sigma$-protocol for the relation:*

$$\mathcal{R} = \{((C_1, C_2, g, h, q), (z)) \mid C_1 C_2^{-1} = h^z\}.$$

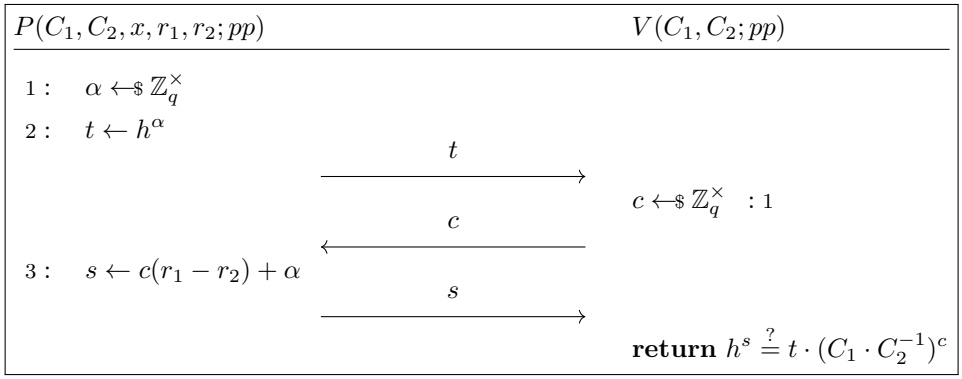



Figure 3.2: Sigma protocol for proving Pedersen commitments open to the same value.

## 3.3 Proving Multiplicative Relations between Commitments

Given $C_1 = \mathsf{Commit}(x) = g^x h^{r_1}$, $C_2 = \mathsf{Commit}(y) = g^y h^{r_2}$, $C_3 = \mathsf{Commit}(z) = g^z h^{r_3}$, A prover may engage in Fig. 3.3 to convince a verifier that $z = xy$.

**Theorem 5.** *Fig. 3.3 is a $\Sigma$-protocol for the relation:*

$$\mathcal{R} = \{((C_1, C_2, C_3, g, h, q), (x, y, r_1, r_2, r_3)) \mid C_1 = g^x h^{r_1},\ C_2 = g^x h^{r_2}, C_3 = g^{xy} h^{r_3}\}.$$

*Proof.* Follows from [WTs+18, Thm. 10]. □

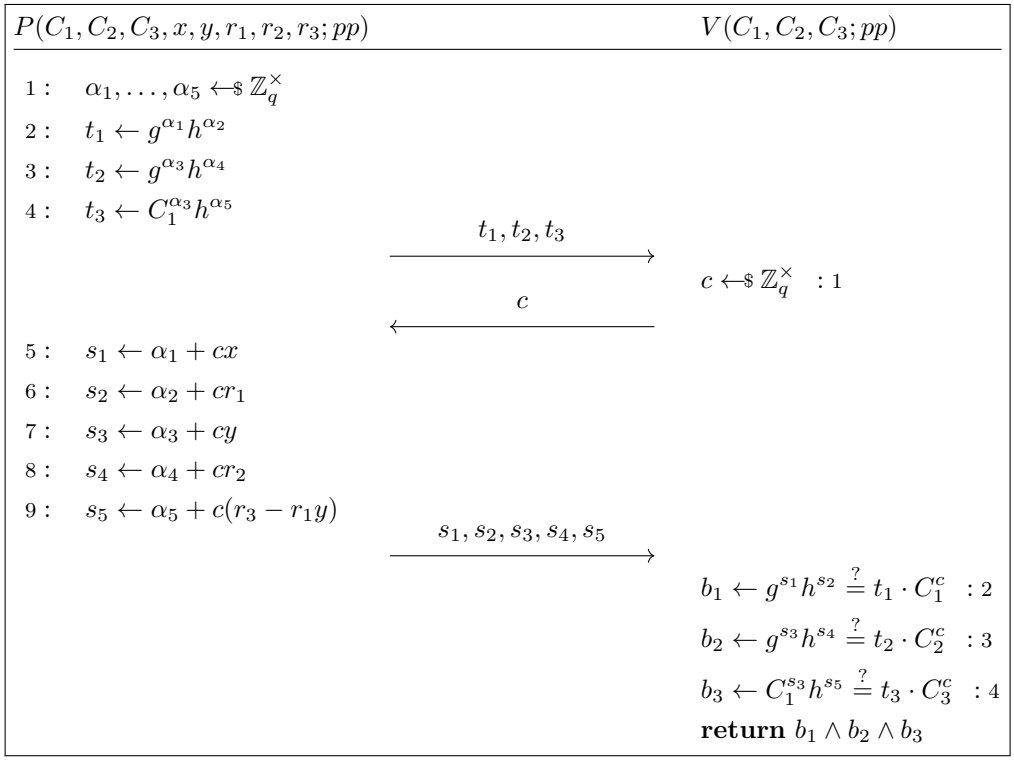



Figure 3.3: Sigma protocol for proving multiplicative relationships between Pedersen commitments. We denote a parallel composition of this proof by calling MulProof with the prover's input.

## 3.4 Proving a Commitment Opens to a Non-Zero Value

Given a commitment $C = \mathsf{Commit}(x) = g^x h^r$, a prover may engage in Fig. 3.4 to convince a verifier that $x \neq 0$.

**Theorem 6.** *Fig. 3.4 is a $\Sigma$-protocol for the relation:*

$$\mathcal{R} = \{(C, (x, r)) \mid C = \mathsf{Commit}(x, r),\ x \neq 0\}.$$

*Proof.* **Completeness.** Observe that $x \neq 0 \implies t_1 \neq 1$ and the other verification equations hold. Hence, $V$ accepts an honest $P$ with probability 1.

**Special Soundness.** Suppose $(\mathtt{comm}, \mathtt{chall}, \mathtt{resp})$, $(\mathtt{comm}', \mathtt{chall}', \mathtt{resp}')$ are two accepting transcripts where $\mathtt{comm} = \mathtt{comm}'$, and $\mathtt{chall} \neq \mathtt{chall}'$. Denote $\mathtt{resp}, \mathtt{resp}'$ as

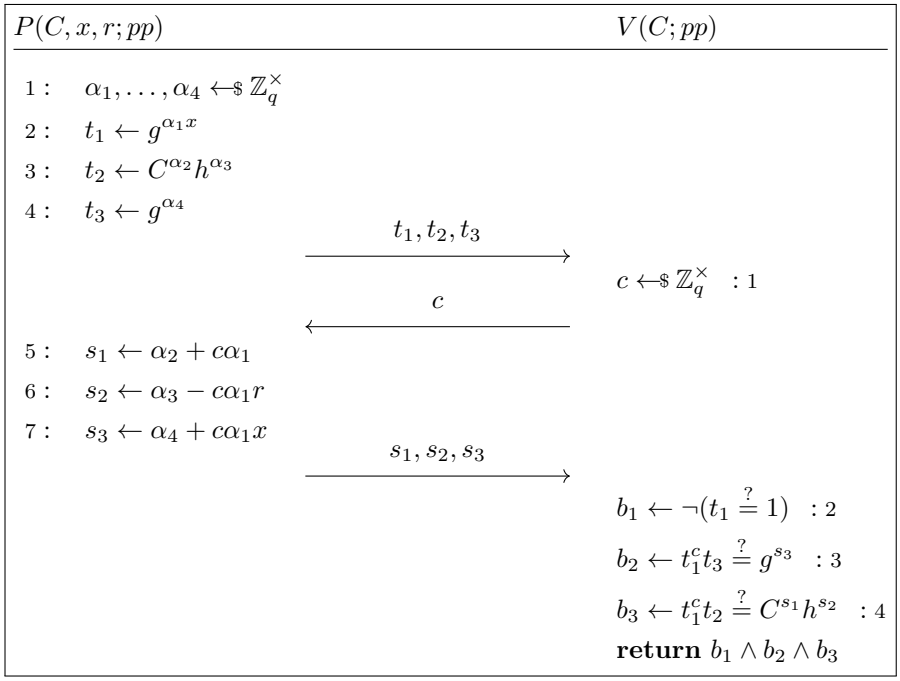



Figure 3.4: Sigma protocol for proving a Pedersen commitment opens to a non-zero value. We denote a parallel composition of this proof by calling NonZeroProof with the prover's input.

$(s_1, s_2, s_3)$ and $(s_1', s_2', s_3')$, respectively. We define the output of an extractor to be $(\alpha_1', x', r')$ where:

$$\alpha_1' = \frac{s_1 - s_1'}{c - c'} \qquad x' = \alpha_1'^{-1} \frac{s_3 - s_3'}{c - c'} \qquad r' = \alpha_1'^{-1} \frac{s_2' - s_2}{c - c'}$$

We argue $x', r'$ is a valid witness to the relation in Theorem 6. First, given the following equations:

$$t_1^c t_3 = g_3^s \qquad \text{and} \qquad t_1^{c'} t_3 = g^{s_3'}$$

we divide the first by the latter, obtaining

$$\begin{aligned} t_1^{c-c'} &= g^{s_3 - s_3'} \\ \implies \qquad t_1 &= g^{(s_3 - s_3')/(c-c')} \end{aligned} \tag{3.1}$$

which implies that the discrete logarithm of $t_1$ is $(s_3 - s_3')/(c - c')$. Second, we consider the equations:

$$t_1^c t_2 = C^{s_1} h^{s_2} \qquad \text{and} \qquad t_1^{c'} t_2 = C^{s_1'} h^{s_2'}.$$

By dividing the two equations, we obtain

$$\begin{aligned} t_1^{c-c'} &= C^{s_1-s_1'} h^{s_2-s_2'} \\ \implies \qquad t_1 &= C^{\alpha_1'} h^{(s_2-s_2')/(c-c')}. \end{aligned} \tag{3.2}$$

By Eqs. (3.1) and (3.2), first we prove that $\alpha_1' \neq 0$ (i.e. $s_1, s_1'$ are distinct). If $\alpha_1' = 0$, then

$$g^{(s_3-s_3')/(c-c')} = h^{(s_2-s_2')/(c-c')}$$

Hence, either the discrete logarithm relation between $g$ and $h$ may be recovered, or both $s_2 = s_2'$ and $s_3 = s_3'$. The former case contradicts the hardness assumption, and the latter case implies that $t_1 = 1$, which contradicts the first verification equation. Therefore, it must be the case that $\alpha_1' \neq 0$, and we have the following:

$$\begin{aligned} & C^{\alpha_1'} h^{(s_2-s_2')/(c-c')} = g^{(s_3-s_3')/(c-c')} \\ \implies \qquad & C^{\alpha_1'} = g^{(s_3-s_3')/(c-c')} h^{(s_2'-s_2)/(c-c')} \\ \implies \qquad & C = g^{\alpha_1'^{-1}(s_3-s_3')/(c-c')} h^{\alpha_1'^{-1}(s_2'-s_2)/(c-c')} \\ \implies \qquad & C = g^{x'} h^{r'}. \end{aligned}$$

Last, we show that if $x' = 0$, the first verification equation cannot be satisfied. Observe that

$$t_1 = g^{(s_3-s_3')/(c-c')} = g^{\alpha_1' x'} = g^0 = 1$$

Hence, the extractor's output is valid.

**Honest Verifier Zero-Knowledge.** On input `chall`, for a given commitment $C$, the simulator $S$ does the following:

1. Samples uniformly random values $s_1, s_2, s_3, \alpha \leftarrow\!\!\$\, [q-1]$, then computes:

$$t_1 = g^{\alpha} \qquad t_2 = \frac{C^{s_1} h^{s_2}}{t_1^c} \qquad t_3 = \frac{g^{s_3}}{t_1^c}.$$

2. Outputs the transcript $((t_1, t_2, t_3), c, (s_1, s_2, s_3))$.

By the choice of $t_1, t_2, t_3$, the transcript satisfies the verification equations above. We show that over uniformly distributed challenges, $S$ outputs transcripts which are identically distributed with transcripts between an honest $P$ and $V$. Fix an instance $C$ and challenge $c$. Observe that a real protocol execution is determined by $P$'s random coins $\alpha_1, \ldots, \alpha_4$, while, on the other hand, a simulated transcript is uniquely determined

by $S$'s random coins $s_1, s_2, s_3, \alpha$. Since both are sampled from the same probability space, and uniquely determine the resulting transcripts, we show that there exists bijection from the random coins of an honest $P$ to the random coins of $S$ that yields identical transcripts for a fixed instance challenge pair. Consider the bijection:

$$(\alpha_1, \alpha_2, \alpha_3, \alpha_4) \mapsto (\alpha_2 + c\alpha_1, \alpha_3 - c\alpha_1 r, \alpha_4 + c\alpha_1 x, \alpha_1 x)$$

By inspection, this map is both injective and surjective when $x \neq 0$ (and, hence, bijective), and when $S$ picks random coins $(\alpha_2 + c\alpha_1, \alpha_3 - c\alpha_1 r, \alpha_4 + c\alpha_1 x, \alpha_1 x)$, it produces a transcript identical to that of an honest $P$ who uses random coins $(\alpha_1, \alpha_2, \alpha_3, \alpha_4)$. Both of these events occur with equal probability. Therefore, over every possible instance, the simulator produces transcripts identical to that of honest protocol executions. □

## 3.5 Proving Generic Statements using Sigma protocols for Pedersen Commitments

We briefly describe an approach to construct a proof system for R1CS over $\mathbb{F}_q \cong \mathbb{Z}_q$ using only Pedersen commitments over a group of order $q$[1], along with Figs. 3.1 and 3.3. There are several other works which introduce general frameworks for proving statements on Pedersen commitments [NBMV99, CM99, AGM18, Bra97]. While this construction could be considered folklore, it is not widely discussed in the context of R1CS, since in most cases the use of *succinct* proof systems is more efficient. For example, one could employ Bulletproofs [BBB+18], which are particularly efficient and operate in the discrete logarithm setting. However, for small arithmetic circuits, and in a classical setting, this approach may be useful. Recall that

$$\mathcal{R}_{\text{R1CS}} = \{(A, B, C, \mathbf{x}, q), (\mathbf{w}) \mid A\mathbf{z} \circ B\mathbf{z} = C\mathbf{z}, \mathbf{z} = (1 \ \mathbf{v} \ \mathbf{w})\}$$

for matrices $A, B, C \in \mathbb{Z}_q^{m \times n+1}$, and vector $\mathbf{z} = (1 \ \mathbf{v} \ \mathbf{w}) \in \mathbb{Z}_q^{n+1}$. Equivalently, we can represent the R1CS as the following system of $m$ equations in $n$ variables. For $j$ in $[m]$:

$$\left(\sum_{i \in [n]} A_{i,j}\mathbf{z}_i,\right) \circ \left(\sum_{i \in [n]} B_{i,j}\mathbf{z}_i\right) = \sum_{i \in [n]} C_{i,j}\mathbf{z}_i. \tag{3.3}$$

In order to prove knowledge of a satisfying assignment $\mathbf{z}$, we can use the following approach:

1. The prover commits to the values $\mathbf{z}$ using Pedersen commitments[2], such that $(P_i = \mathsf{Commit}_q(\mathbf{z}_i))_{i \in [n]}$, and sends the commitments to the verifier.

[1] We note that this requires $q$ to be of sufficient size to guarantee the hardness of computing discrete logarithms in the group of order $q$

[2] for efficiency sake, they could simply set the commitments to the public values as $g^{\mathbf{z}_i}$

2. Observe that the verifier can compute linear combinations of committed values, since the commitments are additively homomorphic. The verifier computes the corresponding commitments to the linear combinations:

$$L_j = \prod_{i\in[n]} P_i^{A_{i,j}}, \qquad R_j = \prod_{i\in[n]} P_i^{B_{i,j}}, \qquad O_j = \prod_{i\in[n]} P_i^{C_{i,j}} \qquad \text{for } j \in [m],$$

where taking the product corresponds to the group operation on the commitments.

3. Let $r_i$ be the associated randomness for the $i$-th commitment $P_i$. The prover computes the following linear combinations, for $j \in [m]$:

$$\alpha_{1,j} = \sum_{i\in[n]} A_{i,j}\mathbf{z}_i, \qquad \alpha_{2,j} = \sum_{i\in[n]} B_{i,j}\mathbf{z}_i,$$
$$\beta_{1,j} = \sum_{i\in[n]} A_{i,j}r_i, \qquad \beta_{2,j} = \sum_{i\in[n]} B_{i,j}r_i, \qquad \beta_{3,j} = \sum_{i\in[n]} C_{i,j}r_i.$$

4. The prover engages in the following sigma protocols in parallel (with the same challenge $c \leftarrow\!\!\$\, \mathbb{Z}_q$):

$$\mathsf{MulProof}(L_j, R_j, O_j, \alpha_{1,j}, \alpha_{2,j}, \beta_{1,j}, \beta_{2,j}, \beta_{3,j}) \text{ for } j \in [m],$$
$$\mathsf{OpeningProof}(P_i, \mathbf{z}_i, r_i) \text{ for } i \in [n].$$

Note that the multiplication proofs prove precisely that the equations in Eq. (3.3) hold, and the opening proofs are needed in order to extract the witness. We note that if the R1CS instance has $n$ variables and $m$ constraints, the proof contains $(3m + 2n)$ commitments and $(5m + n)$ $\mathbb{Z}_q$-elements. The prover performs $(6m + 3n)$ group exponentiations, while the verifier performs $(9m + 2n + O(mn))$ group exponentiations[3], $(6m+2n)$ group operations and $n$ inversions. By applying the Fiat-Shamir transform, the protocol can be made non-interactive. Depending on your requirements, for small R1CS instances this can be quite reasonable. For example, if $m = n = 50$, assuming $\log q = 256$ and commitments are made with 257-bit compressed elliptic curve representations, the proof size is approximately 17 kB.

[3]We remark that the bi-variate term $O(mn)$ is $3mn$ in the worst case if the matrices have all non-zero entries, but is often negligible in practice since the matrices $A, B, C$ are generally sparse (and often also have small entries).

# Chapter 4

# Proving Isogeny Relations with Generic Statements

While attacks have rendered the SIDH assumption broken [MMP+23, CD23, Rob23], the more general $\ell$-isogeny path finding problem remains secure, and underpins the security of most isogeny-based cryptographic schemes. In this section, we present a method to prove the knowledge of several isogeny relations which are useful in the context of constructing isogeny-based protocols. We first construct R1CS instances in Section 4.1 which encode knowledge of a path in the supersingular 2-isogeny graph using modular polynomials. In particular, we aim to construct a proof of knowledge the following relation, which is based on the hardness of the $\ell$-isogeny path finding problem:

$$\mathcal{R}_{\ell^k\text{-IsoPath}} = \{((E_0, E_1), \phi) \ : \ \phi : E_0 \to E_1 \text{ is an isogeny, } \deg \phi = \ell^k, k \in \mathbb{Z}\} \quad (4.1)$$

We then show that radical isogeny formulae can encode isogeny paths more efficiently in Section 4.2. Using standard techniques, we can encode the arithmetic over $\mathbb{F}_{p^2}$ in $\mathbb{F}_p$ with minimal overhead in Section 4.3.

Lastly we introduce a variant of the CGL hash function, and show to prove knowledge of a preimage, as well as proving consistency of parallel evaluations of the function in Sections 4.4 and 4.5. This is useful in the context of our construction presented in Section 5.3.

We summarise the results of applying different techniques to construct R1CS systems for different isogeny relations from this section in Table 4.1.

**High-Level Overview** The reader might wonder, what in particular makes generic proof systems for arithmetic circuits so amenable to isogenies? Firstly, previous PoKs for Relation 4.1 based on sigma protocols [JD11, DDGZ22, GPS17, BCC+23] suffer from various deficiencies. They require computing large degree coprime $N'$-isogenies,

| | | Relation 4.1 | | Relation 4.10 | | Relation 4.19 | |
|---|---|---|---|---|---|---|---|
| | | $n$ | $m$ | $n$ | $m$ | $n$ | $m$ |
| ModPoly-R1CS | $\mathbb{F}_p$ | $11k+7$ | $11k+5$ | **?** | **?** | **?** | **?** |
| | $\mathbb{F}_{p^2}$ | $4k+3$ | $4k-2$ | **?** | **?** | **?** | **?** |
| Radical R1CS | $\mathbb{F}_p$ | $5k+4$ | $5k$ | $12k+4$ | $13k$ | $23k+8$ | $25k$ |
| | $\mathbb{F}_{p^2}$ | $2k+2$ | $2k$ | **?*** | **?*** | **?*** | **?*** |

Table 4.1: Comparison of R1CS systems for different isogeny relations over $\mathbb{F}_p$ and $\mathbb{F}_{p^2}$ for $\ell = 2$. $k$ is the length of the isogeny path, and $n$ and $m$ denote the number of constraints and variables respectively of the R1CS instance over the respective field. '**?**' denotes that the R1CS system was not implemented (but may be possible). *We are not aware of a technique to efficiently canonicalise choice of square root over $\mathbb{F}_{p^2}$.

which means they must either work over larger field extensions or choose a prime large enough to have $\mathbb{F}_{p^2}$-rational $N'$-torsion. They are extremely inefficient due to their small challenge space, resulting in many parallel repetitions to achieve negligible soundness error. Thirdly, the special soundness extractors of these protocols for Relation 4.1 typically recover a degree $N'^2\ell^k$ isogeny, rather than an isogeny of degree $\ell^k$. Hence, we try a different approach. Fortunately, deciding knowledge of an isogeny path can be represented in a very low depth, regular circuit. That is, an arithmetic circuit $C$ where $C(x, w) = 1$ if and only if $(x, w) \in \mathcal{R}_{\ell^k\text{-IsoPath}}$. 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 generic proof systems is as follows:

1. Encode the relation $\mathcal{R}$ and pair $(x, w)$ into an equivalent R1CS, denoted by $\mathcal{R}'$ and $(x', w')$ respectively.

2. Use a generic zk-SNARK for R1CS (resp. arithmetic circuits) to argue the knowledge of a witness $w'$ such that $(x', w') \in \mathcal{R}'$.

3. The prover's knowledge of $w'$ will imply the knowledge of $w$ such that $(x, w) \in \mathcal{R}$.

**Statement of Authorship Contribution (Chapter 4)** *The following section is based on the joint works [CLL23, LP25], and some content is also present in the PhD thesis of Lai. All of the content included in this chapter which was not in my own words has either been rewritten or expanded upon, and I am responsible for the majority of the intellectual contributions of the content present in this chapter. The notable*

*sections of this chapter, of which the coauthors of the prior works were more involved in the intellectual contributions, are: (1) Section 4.3, which was originally written by Lai and conceived along with myself, and was included in Lai's PhD thesis, but has been generalised to handle more arbitrary constraints for this work by the author; and (2) Sections 4.2 and 4.4, of which Pedersen introduced the idea of investigating radical isogenies and supported theoretical discussion. I would like to emphasise that the chapter consists of a collection of results and techniques which are common to these two papers, instigated primarily by myself.*

## 4.1 R1CS for Isogeny Paths from Modular Polynomials

In this section we provide an approach to proving Relation 4.1 via representing knowledge of an isogeny path as knowing the solutions to a system of coupled modular polynomials, an approach first introduced by the verifiable delay function of [CSRT22]. For prime-power degree isogenies, the witness $\phi$ is typically represented by fixing a basis of the $\ell^k$-torsion group, and giving a kernel generator, a point on $E_0$ of order $\ell^k$. Instead, we choose to represent our witness isogeny $\phi$ in the relation above by a sequence of $j$-invariants, encoding a path in the supersingular $\ell$-isogeny graph. Recall, two elliptic curves $E, E'$ are $\ell$-isogenous if and only if $\Phi_\ell(j(E), j(E')) = 0$. Then an isogeny $\phi : E_0 \to E_1$ of degree $\ell^k$ can equivalently be represented as a sequence of intermediate $j$-invariants $j_1, j_2, ..., j_{k-1}$ such that

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

We note that the equations do not impose any restrictions on the underlying finite field. Hence, our method can apply to the CSIDH setting where $j_i$ are defined over the prime field as long as the proof system supports the form of the prime. Our method can complement the efficient proof systems [DG19, BKV19] which have no restrictions on the degree of the witness.

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 $p, \ell, k$. 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 quadratic constraint on the variables. One of these rows must encode the modular polynomial of level $\ell$, $\Phi_\ell(j_i, j_{i+1}) = 0$, which shows that two adjacent $j$-invariants are isogenous. First, we note that

$$\begin{aligned}\Phi_2(X,Y) = X^3 + Y^3 - 162000(X^2+Y^2) - X^2Y^2 + 40773375XY \\ + 1488XY(X+Y) + 8748000000(X+Y) - 157464000000000 \quad (4.2)\end{aligned}$$

For representation compactness, we arrange the statement $\Phi_2(X,Y) = 0$ in the following form:

$$\begin{aligned}-1488XY(X+Y-1488^{-1}XY) = X^3 + Y^3 - 162000(X^2+Y^2) + \\ 8748000000(X+Y) + 40773375XY - 157464000000000 \quad (4.3)\end{aligned}$$

**R1CS over $\mathbb{F}_{p^2}$** We 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.

Suppose the isogeny path in question is of length $k$. If $k = 1$, $\ell = 2$ then by Eq. (4.3), 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

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

where the $c_i$'s are derived from Eq. (4.3). 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. (4.3). Now we can 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 secret $j$-invariant in the sequence[1] 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.

**Adding Requirement for Non-Backtracking** We first note that a non-backtracking isogeny path corresponds directly to a cyclic isogeny (see Section 2.3.1). Hence we show how to prove $\mathcal{R}_{\ell^k\text{-CyclicIsoPath}}$ (the same as Relation 4.1 with the added requirement that the witness is a cyclic isogeny). In the modular polynomial relation we introduce, we do not provide any guarantee that our isogeny is non-backtracking (and hence cyclic).

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 an 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. In other words, 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 optimisation 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 $\delta_i$, 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 $\delta'$ such that $\delta\delta' = 1$,

where the $\delta$ term will be non-zero if and only if all $\delta_i$ are non-zero, which is true if (but not only if) the walk is non-backtracking (see Remark 2).

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.

*Remark* 2. We note that this check will also prevent the use a 2-cycle (with two distinct edges, corresponding to a norm 4 endomorphism), which is still a cyclic isogeny. So while technically the R1CS fails to prove cyclic isogenies which contain a 2-cycle, we

[1]The verifier can compute the squarings and cubings of $j_0$ and $j_k$ offline and input them directly, hence the checks are not required.

stress these exceptional cases are extremely rare. In particular, there are at most 5 pairs of double edges in the $G_2(p)$ [ACNL+21, Cor. 2.15] out of approximately $\frac{p}{8}$ edges (by the handshaking lemma). However, depending on the application, a cyclic $\ell^k$-isogeny containing a 2-cycle may be translated into a cyclic $\ell^{k-2}$-isogeny, which may still be acceptable; and another approach is to work over a prime $p = 1 \pmod{420}$, which does not possess 2-cycles [CLG09]. Nevertheless, this issue is completely circumvented by the techniques introduced in the next section. This is another advantage of the radical isogeny formulae, which by design only produces cyclic isogenies, and hence does not require this check.

## 4.2 R1CS for Isogeny Paths from Radical Isogeny Formulae

In this section, we develop a more efficient R1CS description which encodes knowledge of a $2^k$-isogeny for any $k$ as in Section 4.1, but rather than using modular polynomials, we apply the radical isogeny formulas from [CDV20]. Due to the requirements of the radical isogeny formulae, we now restrict ourselves to the setting of computing isogenies over $\mathbb{F}_{p^2}$, $p \equiv 3 \pmod 4$ (which is standard for isogeny-based cryptography), and curves represented via the equation

$$y^2 = x^3 + Ax^2 + Cx, \tag{4.4}$$

where $A, C \in \mathbb{F}_{p^2}$. Results from the prior section do not have this restriction, but we stress that the R1CS can also be modified to input public $j$-invariants $j_0$, $j_k$ as a representation for the starting and end curve. See Remark 3.

**Radical Isogeny Walks.** Rather than using the 2-modular polynomial, we encode knowledge of an isogeny path by using the *radical isogeny* formulas from [CDV20] (we note that the formula for $\ell = 2$ was already been proposed in [CD20]). In their work, the authors introduce a faster way to compute 2-isogenies, which removes the need to generate the 2-torsion in the first place, thus speeding up the computation. The dominant part of the isogeny computation then becomes taking a square-root. An isogeny of degree 2 connecting two curves $E_i$ and $E_{i+1}$ can then be expressed as

$$E_i : y^2 = x^3 + A_i x^2 + C_i x \quad \longrightarrow \quad E_{i+1} : y^2 = x^3 + A_{i+1} x^2 + C_{i+1} x\,,$$

where the coefficients are connected via the following simple formulas

$$A_{i+1} = 6\sqrt{C_i} + A_i\,, \quad C_{i+1} = 4\sqrt{C_i} A_i + 8C_i\,. \tag{4.5}$$

By substituting the first equation into the second, and adding a squaring constraint, we can encode knowledge of an isogeny path of length $k$ in the 2-isogeny graph (i.e. an instance of Relation 4.1) in an R1CS instance which has the constraints:

$$6C_{i+1} - 48C_i = 4A_i(A_{i+1} - A_i) \tag{4.6}$$

$$36 \cdot C_i = (A_{i+1} - A_i)^2 \tag{4.7}$$

for $i = 0, \ldots, k-1$. Previously, the constraints required cubic terms in the $j$-invariants of elliptic curves, and now the equations are solely quadratic. This yields an R1CS instance with $2k+2$ variables ($A_i, C_i$ for $i = 0, \ldots, k$) and $2k$ constraints (see above), rather than the $4k+3$ variables and $4k-2$ constraints from the modular polynomial approach. We note that the radical isogeny formulas are by design non-backtracking, so we do not need to take extra care to ensure the isogenies are cyclic. More precisely, this R1CS instance is a proof of knowledge for the relation $\mathcal{R}_{\ell^k\text{-CyclicIsoPath}}$.

*Remark* 3. Suppose one wishes to prove Relation 4.1 where the public input is represented by two $j$-invariants $j_0$ and $j_k$, and witness is a path of $k$ $j$-invariants:

$$j_0, j_1, \ldots, j_{k-1}, j_k.$$

We show how to modify the radical R1CS construction to accommodate this. First, the prover needs to perform some precomputation. First, they choose a suitable model for $E_0 : y^2 = x^3 + A_0x^2 + C_0x$ such that $j(E_0) = j_0$ and such that the outgoing two curves computable by radical isogeny formulae includes a curve of $j$-invariant $j_1$. At each of the $i \in [k]$ steps, the prover can then determine which of the possible outgoing curves $E_{i+1}$ (corresponding to the choice of $\sqrt{C_i}$ in Equation (4.5)) has $j(E_{i+1}) = j_{i+1}$, and then use the sequence of curve coefficients $\{A_i, C_i\}_{i\in[k+1]}$ as secret input to the R1CS. The verifier still needs to be convinced that the $j_0 = j(E_0)$ and $j_k = j(E_k)$ match the public input. While the prover can simply publish the curve coefficients $A_0, C_0$ and $A_k, C_k$, which the verifier can then check offline, it may be the case that the curve equations leak some information about the secret path. Hence, the prover may wish to assert the correctness of the $j$-invariants $j_0, j_k$ within the R1CS instance. Using the fact that for this model of curve

$$j(E_i) := 256 \cdot \frac{(A_i^2 - 3C_i)^3}{C_i^2(A_i^2 - 4C_i)},$$

one can add the following extra constraints to the R1CS for $i \in \{0, k\}$:

$$\begin{aligned}
A_i^2 = \gamma_i, \qquad & C_i^2 = \delta_i, && \text{(to verify the squarings)} \\
(\gamma_i^2 - 3C_i)^2 = \epsilon_i, \quad & \epsilon_i(\gamma_i^2 - 3C_i) = \eta_i, && \text{(for the numerator)} \\
\delta_i(\gamma_i - 4C_i) = \zeta_i, \qquad & j_i \cdot \zeta_i = 256 \cdot \eta_i && \text{(for the denominator and final check)}
\end{aligned}$$

This requires only an additional 12 constraints and 12 variables, which is negligible overhead for cryptographic sized $k \geq 2\lambda$.

## 4.3 Transforming $\mathbb{F}_{p^2}$ R1CS instances into $\mathbb{F}_p$

In Sections 4.1 and 4.2, we construct R1CS instances for verifying isogeny paths over the full supersingular isogeny graph (which operates over $\mathbb{F}_{p^2}$). However, it may be desirable, and in some cases necessary (such as in the following section), to embed the R1CS instance into the base field $\mathbb{F}_p$. Since $p \equiv 3 \mod 4$, we may interpret $\mathbb{F}_{p^2}$ as $\mathbb{F}_p[j]$ for $j^2 = -1$. We show how to convert an $\mathbb{F}_p[j]$ R1CS instance into an $\mathbb{F}_p$ R1CS instance below:

**Input.** We encode each non-trivial entry $\mathbf{z}_i$ in the vector $\mathbf{z} := (1||\mathbf{v}||\mathbf{w})$ as a pair of elements $x_i, y_i \in \mathbb{F}_p$ where $\mathbf{z}_i$'s are naturally interpreted as $x_i + y_i j$. We will refer to $\mathsf{Re}(\mathbf{z}_i) = x_i$ as the real part and $\mathsf{Im}(\mathbf{z}_i) = y_i$ as the imaginary part. The first entry of $\mathbf{z}$ is always 1 and does not need an additional variable.

**Squaring.** Each $\mathbb{F}_p[j]$ squaring $\mathbf{z}_a^2 = \mathbf{z}_b$ (indexed by $a, b$), can be encoded using only 2 $\mathbb{F}_p$ constraints:

$$\begin{aligned} 2\mathsf{Re}(\mathbf{z}_a)\mathsf{Im}(\mathbf{z}_a) &= \mathsf{Im}(\mathbf{z}_b) && \text{(for the imaginary part)} \\ (\mathsf{Re}(\mathbf{z}_a) + \mathsf{Im}(\mathbf{z}_a)) \cdot (\mathsf{Re}(\mathbf{z}_a) - \mathsf{Im}(\mathbf{z}_a)) &= \mathsf{Re}(\mathbf{z}_b) && \text{(for the real part)} \end{aligned}$$

It is easy to observe that the above equations hold if and only if $\mathbf{z}_a^2 = \mathbf{z}_b$ in $\mathbb{F}_p[j]$.

**Multiplication.** For each $\mathbb{F}_p[j]$ multiplication $\mathbf{z}_a \mathbf{z}_b = \mathbf{z}_c$, (indexed by $a, b, c$) we encode the following 3 $\mathbb{F}_p$ constraints (requiring 1 additional variable $u_{a,b,c}$):

$$\begin{aligned} \mathsf{Im}(\mathbf{z}_a)\mathsf{Im}(\mathbf{z}_b) &= u_{a,b,c} && \text{(introducing a variable)} \\ \mathsf{Re}(\mathbf{z}_a)\mathsf{Re}(\mathbf{z}_b) &= \mathsf{Re}(\mathbf{z}_c) + u_{a,b,c} && \text{(for the real part)} \\ (\mathsf{Re}(\mathbf{z}_a) + \mathsf{Im}(\mathbf{z}_a)) \cdot (\mathsf{Re}(\mathbf{z}_b) + \mathsf{Im}(\mathbf{z}_b)) &= \mathsf{Im}(\mathbf{z}_c) + \mathsf{Re}(\mathbf{z}_c) + 2u_{a,b,c} && \text{(for the imaginary part)} \end{aligned}$$

**Expressing Arbitrary R1CS Constraints.** We note that expressing any R1CS constraint, which corresponds to an equation with a product of linear expressions on the left and a linear expression on the right, can be done in the same manner as multiplication, provided that the coefficients of the linear expressions are solely in $\mathbb{F}_p$. Observe that the embedding is $\mathbb{F}_p$-linear, i.e. $c_a\mathsf{Re}(\mathbf{z}_a) + c_b\mathsf{Re}(\mathbf{z}_b) = \mathsf{Re}(c_a\mathbf{z}_a + c_b\mathbf{z}_b)$ and $c_a\mathsf{Im}(\mathbf{z}_a) + c_b\mathsf{Im}(\mathbf{z}_b) = \mathsf{Im}(c_a\mathbf{z}_a + c_b\mathbf{z}_b)$ for all $c_a, c_b \in \mathbb{F}_p$. Hence, an arbitrary constraint in $n$ variables over $\mathbb{F}_p[j]$:

$$(\sum_{l=1}^{n} c_l \mathbf{z}_l) \cdot (\sum_{r=1}^{n} d_r \mathbf{z}_r) = \sum_{o=1}^{n} e_o \mathbf{z}_o,$$

where $c_i, d_i, e_i \in \mathbb{F}_p$ and $\mathbf{z}_i \in \mathbb{F}_p[j]$ for all $i \in [n]$, can be encoded using only 3 constraints and 1 additional variable[2] in $\mathbb{F}_p$:

$$\sum_{l=1}^{n} c_l \mathsf{Im}(\mathbf{z}_l) \cdot \sum_{r=1}^{n} d_r \mathsf{Im}(\mathbf{z}_r) = u$$

$$\left(\sum_{l=1}^{n} c_l \mathsf{Re}(\mathbf{z}_l)\right) \cdot \left(\sum_{r=1}^{n} d_r \mathsf{Re}(\mathbf{z}_r)\right) = \sum_{o=1}^{n} e_o \mathsf{Re}(\mathbf{z}_o) + u$$

$$\left(\sum_{l=1}^{n} c_l \mathsf{Re}(\mathbf{z}_l) + c_l \mathsf{Im}(\mathbf{z}_l)\right) \cdot \left(\sum_{r=1}^{n} d_r \mathsf{Re}(\mathbf{z}_r) + d_r \mathsf{Im}(\mathbf{z}_r)\right) = \left(\sum_{o=1}^{n} e_o \mathsf{Im}(\mathbf{z}_o) + e_o \mathsf{Re}(\mathbf{z}_o)\right) + 2u$$

If the R1CS instance over $\mathbb{F}_p[j]$ has

$$v \text{ variables, } s \text{ squaring constraints and } g \text{ general constraints,}$$

then the resulting R1CS instance over $\mathbb{F}_p$ will have:

$$2v + g \text{ variables and } 2s + 3g \text{ constraints.}$$

**Modular Polynomial R1CS over $\mathbb{F}_p$.** Since the modular polynomial R1CS natively has $4k + 3$ variables and $4k - 2$ constraints ($k - 1$ of which are squarings), the resulting R1CS instance over $\mathbb{F}_p$ will have $11k + 7$ variables and $11k - 5$ constraints. Adding non-backtracking checks would result in overall $14k + 7$ variables and $14k - 5$ constraints.

**Radical Isogeny R1CS over $\mathbb{F}_p$.** Since the radical isogeny R1CS natively has $2k + 2$ variables and $2k$ constraints ($k$ of which are squarings), the resulting R1CS instance over $\mathbb{F}_p$ will have $5k + 4$ variables and $5k$ constraints.

## 4.4 Proving Knowledge of a CGL Preimage

In the 2-isogeny graph, every vertex has three outgoing isogenies, so at every step except the first, without backtracking, we have exactly two options to continue our path. In the original CGL construction, the 2-torsion is ordered in some deterministic fashion (where the backtracking edge is disregarded) and the input $m$, read as bits, determines whether to go "left" ($m_i = 0$) or "right" ($m_i = 1$) at every bifurcation.

In this section, we show how to construct an R1CS instance which asserts knowledge of a 2-isogeny walk predetermined by a directional bit-string. Essentially, on public

[2] As a slight abuse of notation, we avoid indexing the variable $u$ here, but it should be unique for each constraint to be embedded into $\mathbb{F}_p$.

input tuple of supersingular curves $(E_0, E_m)$, we prove knowledge of a secret input $m \in \{0,1\}^n$ such that

$$E_m = \mathsf{CGL}(E_0, m).$$

We extend the latter to allow the proof that two different paths have been computed using the same string as an input.

We start this section with a construction for a radical isogeny version of CGL, which is interesting in its own right. We then move on to building a proof system of correct CGL evaluation, and finally simplify/extend this to the above-mentioned proofs.

Clearly, there are two choices for $\sqrt{C_i}$. In fact, in the 2-isogeny graph, this choice corresponds to going left or right in a non-backtracking walk, in a similar way as CGL does. There are many ways to distinguish these two roots, but since we plan to later distinguish them algebraically as part of our R1CS constraints, we use the following method.

Let $\pm\alpha_i$ be the square roots of $C_i$. By working over $p \equiv 3 \pmod 4$, we can distinguish $\pm\alpha_i$ via the residuosity of their real (or imaginary) part. Recall that $-1$ is a non-residue in $\mathbb{F}_p$ for $p \pmod 4$; and since $\mathsf{Re}(\alpha_i) \in \mathbb{F}_p$, we find that, if $\mathsf{Re}(\alpha_i) \neq 0$, precisely one of $\mathsf{Re}(\alpha_i)$ and $\mathsf{Re}(-\alpha_i)$ are quadratic residues and non-quadratic residues respectively. In the case that $\mathsf{Re}(\alpha_i) = 0$, since by construction $C_i \neq 0$, we have $\mathsf{Im}(\alpha_i) \neq 0$. In this case, we can distinguish the two roots by looking at the second coefficient, since $\mathsf{Im}(\pm\alpha_i) = \pm\mathsf{Im}(\alpha_i)$.

From here, if $\mathsf{Re}(\pm\sqrt{C_i}) \neq 0$, we define $\alpha_i$ as the square root of $C_i$ for which $\mathsf{Re}(\alpha_i)$ is itself a square. If $\mathsf{Re}(\pm\sqrt{C_i}) = 0$, we simply take $\alpha_i$ to be the square root for which $\mathsf{Im}(\alpha_i)$ is a square.

We reinterpret the bits of an input $m$ as $m_i \in \{-1, 1\}$. If $m_i = 1$, we walk the path in our bifurcation that is determined by $+\alpha_i$, and if $m_i = -1$, we walk the path determined by $-\alpha_i$. We can rewrite the relation between the curve coefficients (from Equation (4.5)) as

$$A_{i+1} = 6m_i\alpha_i + A_i, \tag{4.8}$$
$$C_{i+1} = 4m_i\alpha_i A_i + 8C_i\,. \tag{4.9}$$

Similarly to CGL, this allows us to feed in an input $m = m_0 m_1 \dots m_{n-1}$, that results in a deterministic path from a starting curve $E_0$ to some output $E_m$. We note that the choice of parameters of the starting curve immediately excludes one possible path using the formulas mentioned above, which correspond to the isogeny generated by the kernel point $(0,0)$, so we do not need to take extra care at this first step other than preventing a double edge at the start of the walk. Furthermore, backtracking is excluded by the nature of the formulas themselves. We summarise our algorithm in Algorithm 1 below.

We are now building a proof system for the following relation

$$\mathcal{R}_{\mathsf{CGL}} = \{((E_0, E_n); m) \mid E_n = \mathsf{CGL}(E_0, m)\}\ , \tag{4.10}$$

**Algorithm 1** $\mathsf{CGL}(E_0, m)$: Novel variant of CGL using radical isogeny formulas

**Require:** Coordinates $(A_0, C_0) \in \mathbb{F}_{p^2}$ defining a supersingular elliptic curve $E_0 : y^2 = x^3 + A_0x^2 + C_0x$, message $m = m_1m_2 \dots m_n \in \{-1, 1\}^n$

**Ensure:** Coordinates $(A_n, C_n) \in \mathbb{F}_{p^2}$ defining a supersingular elliptic curve $E_n : y^2 = x^3 + A_nx^2 + C_nx$

**for** $i = 0$ to $n - 1$ **do**
  $\alpha_i \leftarrow \sqrt{C_i}$ ▷ Start with arbitrary root
  **if** $\mathsf{Re}(\alpha_i) \neq 0$ and $\mathsf{Re}(\alpha_i)$ is not a square **then**
    $\alpha_i \leftarrow -\alpha_i$
  **else if** $\mathsf{Re}(\alpha_i) = 0$ and $\mathsf{Im}(\alpha_i)$ is not a square **then**
    $\alpha_i \leftarrow -\alpha_i$
  **end if**
  $A_{i+1} \leftarrow 6m_i\alpha_i + A_i$
  $C_{i+1} \leftarrow 4m_i\alpha_iA_i + 8C_i$
**end for**
**return** $(A_n, C_n)$

where we denote $\mathsf{CGL}(E_0, m)$ as the correct output of the algorithm in Algorithm 1 with input some starting curve $E_0$ and a secret input $m = m_0 \dots m_{n-1} \in \{-1, 1\}^n$. The proof system needs to show that Equations (4.8) and (4.9) has been computed correctly, using the correct direction $m_i$ at every step.

We describe our proof system as a Rank-1 Constraint System (R1CS). The easiest way to prove that an element of $\mathbb{F}_p$ is indeed a square is to again provide the square root, which we will denote as $\beta_i$, an element of $\mathbb{F}_p$. In order to account for the correct choice of $\alpha_i$, we introduce a conditional variable $b_i \in \{0, 1\}$, which will need to be zero if $\mathsf{Re}(\alpha_i) \neq 0$, and one otherwise. For $\beta_i$, we do not need to distinguish which root we are talking about, only that $\mathsf{Re}(\alpha_i) + b_i\mathsf{Im}(\alpha_i)$ has a root. Together with the radical isogeny formulae, we find the following system of equations

$$\beta_i^2 = \mathsf{Re}(\alpha_i) + b_i\mathsf{Im}(\alpha_i) \tag{4.11}$$
$$\alpha_i^2 = C_i \tag{4.12}$$
$$A_{i+1} - A_i = 6m_i\alpha_i \tag{4.13}$$
$$C_{i+1} - 8C_i = 4m_i\alpha_iA_i\,. \tag{4.14}$$

By plugging in Eq. (4.13) into Eq. (4.14) and scaling, we can reduce the cubic term in Eq. (4.14) to a simple multiplication

$$3C_{i+1} - 24C_i = 2A_i(A_{i+1} - A_i)\,, \tag{4.14}$$

removing the need for an extra constraint.

Since we want to distinguish square roots via their residuosity, we need to reinterpret[3] $\mathbb{F}_{p^2} = \mathbb{F}_p(j)$ as $\mathbb{F}_p \times \mathbb{F}_p$. First, we note that Equation (4.11), is written as two constraints over $\mathbb{F}_p$ (due to the two quadratic terms in $\mathbb{F}_p$ elements) and can be expressed by introducing the variable $t_i \in \mathbb{F}_p$:

$$t_i = b_i \mathsf{Im}(\alpha_i)$$
$$\beta_i^2 = \mathsf{Re}(\alpha_i) + t_i$$

We can write the squaring Equation (4.12) over $\mathbb{F}_{p^2}$ as two constraints over $\mathbb{F}_p$, while the multiplication in Equation (4.14) can be written as three constraints, using an auxiliary value (see Section 4.3). Note that the multiplication in Equation (4.13) is a multiplication of a scalar with an element from $\mathbb{F}_p$, and thus can be written as two constraints:

$$\mathsf{Re}(A_{i+1}) - \mathsf{Re}(A_i) = 6 m_i \mathsf{Re}(\alpha_i)$$
$$\mathsf{Im}(A_{i+1}) - \mathsf{Im}(A_i) = 6 m_i \mathsf{Im}(\alpha_i)$$

We are not done yet, since we need to prove consistency of the elements $m_i$ and $b_i$. First, we have to ensure $m_i \in \{-1, 1\}$, and $b_i \in \{0, 1\}$, which can be done with the constraints over $\mathbb{F}_p$:

$$(m_i + 1)(m_i - 1) = 0 \tag{4.15}$$
$$b_i(b_i - 1) = 0 \tag{4.16}$$

which is true if and only if $m_i \in \{-1, 1\}$ and $b_i \in \{0, 1\}$. Finally, we also need to ensure that $b_i = 1$ if and only if $\mathsf{Re}(\alpha_i) = 0$, and $b_i = 0$ otherwise. This can be done by adding the following constraints over $\mathbb{F}_p$ (with an additional variable $\iota_i$):

$$b_i \mathsf{Re}(\alpha_i) = 0 \tag{4.17}$$
$$\big(b_i - \mathsf{Re}(\alpha_i)\big) \cdot \iota_i = 1 \tag{4.18}$$

The first constraint ensures that at least one of $b_i$ or $\mathsf{Re}(\alpha_i)$ is zero; and the second ensures that $b_i \neq \mathsf{Re}(\alpha_i)$, since their difference has an inverse $\iota_i$. Summing up, for a path of length $n$, we have the variables:

$$\mathsf{Re}(A_i), \mathsf{Im}(A_i), \mathsf{Re}(C_i), \mathsf{Im}(C_i) \text{ for } i = 0, \ldots, n$$

$$\text{and } \mathsf{Re}(\alpha_i), \mathsf{Im}(\alpha_i), \beta_i, m_i, b_i, t_i, \iota_i, u_i \text{ for } i = 0, \ldots, n-1,$$

[3] An arithmetic circuit for encoding the function $\mathsf{Re}(\cdot)$ would be prohibitively expensive (relying on the frobenius identity $x^p = x$ if.f. $x \in \mathbb{F}_p$)

where $u_i$ is the intermediate variable required in Equation (4.14). For each $i = 0, \ldots, n-1$, we have 2 constraints for Equation (4.11), 2 constraints for Equation (4.12), 2 constraints for Equation (4.13), 2 constraints for Equation (4.14) and one constraint for each of Equations (4.15) to (4.18). Hence the constraint system amounts to

$$12n + 4 \text{ variables and } 12n \text{ constraints.}$$

## 4.5 Proof of same CGL input.

For some applications (hinting towards Section 5.2), one might want to prove that two paths in the 2-isogeny graph have been computed using the same (secret) input $m$, starting from different elliptic curves. Let for example $(E_1, E_2)$ and $(F_1, F_2)$ be two tuples of elliptic curves, then we can define the corresponding relation as

$$\mathcal{R}_{\mathsf{CGL}/\!/} = \{((E_1, E_2, F_1, F_2); m) \mid E_2 = \mathsf{CGL}(E_1, m) \wedge F_2 = \mathsf{CGL}(F_1, m)\} \ . \tag{4.19}$$

This implies that we need a constraint system that realises Equations (4.11) to (4.18) twice in parallel, with the same $m$ as input. This means our R1CS will consist of the constraints and variables from both computations, but with the same input $m$ and one common check of Equation (4.15). In total, this simple modification results in a proof system with

$$23n + 8 \text{ variables and } 23n \text{ constraints}$$

over $\mathbb{F}_p$. We denote running this proof as $\pi \leftarrow \mathsf{NIZK}^{/\!/}.P(m, (E_1, E_2, F_1, F_2))$, while verifying it as $0/1 \leftarrow \mathsf{NIZK}^{/\!/}.V(\pi, (E_1, E_2, F_1, F_2))$.

## 4.6 Proof Systems for R1CS

At the time of writing [CLL23], the state-of-the-art proof system for R1CS operating over FFT-friendly fields was Aurora [BCR+19]. However, the area has advanced significantly

| | Relation 4.1 | Relation 4.10 | Relation 4.19 |
|---|---|---|---|
| Instance Size | $< 2^{11}$ | $< 2^{12}$ | $< 2^{13}$ |
| Prover Time (ms) | 25 | 45 | 75 |
| Verification Time (ms) | 15 | 20 | 25 |
| Proof size (kB) | 230 | 320 | 430 |

Table 4.2: Rough performance estimates obtained from the proof system in [BFK+24, Fig. 2] on the Radical isogeny R1CS instances over $\mathbb{F}_p$ (third row of Table 4.1). We set $\lambda = 128$, and hence $k = 256$, and $\log_2 p = 256$.

since early 2023, and as of 2024, there are now numerous candidates for the state-of-the-art [XZS22, ZCF24, BFK+24, GLH+25, ACFY25]. Several of which are *field agnostic*, they do not impose any conditions on the factors of $\mathbb{F}_q^\times$. Field agnostic SNARKs are particularly well-suited for our application, since in isogeny-based protocols where $p = f\prod_{i=1}^{n}\ell_i^{e_i} - 1$, it is not generally the case that $|\mathbb{F}_p^\times| = p - 1$ has sufficient 2-adicity (or smoothness in the case of different interpolation domains) for FFT operations in a SNARK. Indeed the case we consider in this work, where $p = 2^a \cdot f - 1$, we have that $p - 1 = 2(2^{a-1} \cdot f - 1)$ which has a 2-adicity of 1. Succinct proof systems are challenging to estimate performance for, especially in the case of isogeny computations, which operate over large prime fields that require big integer arithmetic not natively supported by proof system implementations. Since this would require substantial engineering effort and great care in order to support zero-knowledge and the large field arithmetic, we leave concrete implementation details and more accurate performance evaluations to future work. However, we provide some rough estimates of the performance of the most suitable candidate, [BFK+24]. Conveniently, they provide results for a 256-bit field in their paper, in the context of proving ECDSA circuits. We provide their benchmark results on R1CS instances of the relevant sizes in Table 4.2, which were performed on an AWS c5a.16xlarge Ubuntu 22.04 machine with 64 cores and 124 Gb memory.

*Remark* 4. We note that their implementation does not include zero-knowledge, but can be added using the known transformations of [BCG+17, BCR+19, XZZ+19]. This typically adds a constant overhead to the protocol (e.g. 2-fold increase in instance size). We also note that the performance results of [BFK+24] do not account for the further optimisations present in [DP24], which are expected to reduce prover and verification time by a factor of 2, and reduce proof size by a factor of $\sqrt{2}$. Since this matches the asymptotic growth of doubling the instance size, we therefore expect that this optimisation should account for the performance loss from adding zero-knowledge.

# Chapter 5

# Verifiable Random Functions from Isogenies

A Pseudo-Random Function (PRF) is a keyed function whose outputs are indistinguishable from the uniform distribution on the image set, given that the distinguisher does not know the key. PRFs are a powerful primitive which can be used to construct a wide range of cryptographic protocols. For example, existence of PRFs imply provably secure CPA encryption schemes, pseudorandom generators, and message authentication codes. However, the existence of PRFs is a non-trivial assumption, and constructing them from standard assumptions remains a wide open problem in cryptography.

A stronger requirement is the existence of a PRF that also has an associated efficient proof of evaluation. More precisely, an efficient non-interactive proof of knowledge that proves that $x, y$ is a valid input-output pair for $f_k$, without revealing the key $k$. While it is plausible that PRFs instantiated via cryptographic hash functions are secure, (i.e. setting $f_k(M) = \mathsf{HMAC}(k, m)$ for an appropriate hash-based message authentication code), it is challenging to construct efficient proofs of evaluation. Known approaches using SNARKs (see Chapter 4) are slowly becoming more practical, but the underlying evaluation circuit of a hash function is generally quite large. For example, a single SHA-256 evaluation requires approximately $2^{16}$ binary R1CS constraints [AHIV17]. This motivates our investigation into new PRF assumptions in the post-quantum setting. More precisely, we consider *Verifiable Random Functions* (VRFs), which are PRFs with an associated proof of evaluation, first introduced in [MRV99]. We note their properties, which we discuss in the following section, slightly differ from conventional PRFs.

In this chapter we introduce an approach to construct VRFs from what we will call *unpredictable functions*, a related notion to the functions introduced in the original work of [MRV99], along with a non-interactive zero-knowledge proof of knowledge of evaluation. As a novel contribution, we prove security of this "folklore" construction

in the random oracle model. We note that our construction is very similar to the concurrent work[1] of [GS24], which additionally show that their construction (and by equivalence, our construction) satisfies the stronger property of *unbiasability*. This unbiasability property is a necessary property to ensure fairness in a VRF-based leader election protocol, which was not implied by the existing properties of VRFs.

We show that isogenies are promising candidates for unpredictable functions, and we instantiate our VRF with two isogeny settings: one where the function is based on an optimised CGL hash function [CLG09] using radical isogenies [CDV20], and another based on an SIDH-like construction, where the message input defines the kernel of a $2^a$-isogeny. Both of these protocols rely on ad-hoc "one-more" type computational assumptions which we introduce in their respective sections – essentially, that the proposed isogeny-based functions satisfy the notion of unpredictability, as well as a starting curve of unknown endomorphism ring, which requires trusted setup. We believe that instantiations from other paradigms might be possible, and leave this open for further research.

**Statement of Authorship Contribution (Chapter 5)** *The following section is based on the joint work [LP25]. All of the content included in this chapter which was not in my own words has either been rewritten or expanded upon, and I am responsible for the majority of the intellectual contributions of the content present in this chapter. The security proof and the candidate functions, as well as the motivation of security, was worked on in part by my co-author, Robi Pedersen.*

## 5.1 Verifiable Random Functions

Verifiable random functions were first formalised by Micali, Rabin and Vadhan [MRV99]. They provide a mechanism for a user, with an associated public-private key pair, to generate publicly verifiable randomness. The authors define a verifiable random function (VRF) as a tuple of polynomial time algorithms $\Pi_{\mathsf{VRF}} = (\mathsf{SetUp}, \mathsf{KeyGen}, \mathsf{Eval}, \mathsf{Verify})$ with the properties below. Let $\mathcal{K}$ be the set of valid secret keys, and $\mathcal{M}$ be the set of valid messages/inputs.

- $\mathsf{SetUp}(1^\lambda)$ takes as input a security parameter $\lambda$ and returns public parameters $\mathsf{pp}$.
- $\mathsf{KeyGen}(\mathsf{pp})$ takes as input public parameters $\mathsf{pp}$ and returns a secret-public key pair $(\mathsf{sk}, \mathsf{pk})$.

[1] The authors of [GS24] rely on *Verifiable Unbiasable Functions* (VUFs), which are essentially unpredictable functions with an associated proof, whereas we add the non-interactive proof separately within the transformation. The resulting protocols are equivalent.

- $\mathsf{Eval}_{\mathsf{sk}}(m)$ takes as input a secret key $\mathsf{sk}$ and an input string $m \in \mathcal{M}$, then returns an output value $h$ and a proof $\pi$ of the correctness of $h$.
- $\mathsf{Verify}_{\mathsf{pk}}(m, h, \pi)$ takes as input the public key $\mathsf{pk}$, the output $h$ and proof $\pi$, as well as the input $m$ and returns either 1 or 0, indicating that it accepts or rejects the proof.

The security of VRFs is formalised through three security properties. *Provability* states that any correct evaluation of the VRF should result in an output pair that passes the verification algorithm. *Unique provability* further implies that this output pair is unique, i.e. that for a given input and public key, there do not exist distinct outputs that correctly verify. Finally, any adversary interacting with the VRF-functionality should not be able to find an input-output pair that passes verification. This is formalised in the *residual pseudorandomness* property.

**Definition 12** (Provability)**.** For any input $m \in \mathcal{M}$, a correctly generated evaluation will result in an accepting proof with overwhelming probability. Formally, the following holds:

$$\mathsf{Pr}\left[\mathsf{Verify}_{\mathsf{pk}}(m,h,\pi)=1 \;\middle|\; \begin{array}{c}\mathsf{pp}\leftarrow\mathsf{SetUp}(1^\lambda)\\ (\mathsf{sk},\mathsf{pk})\leftarrow\mathsf{KeyGen}(\mathsf{pp})\\ (h,\pi)\leftarrow\mathsf{Eval}_{\mathsf{sk}}(m)\end{array}\right] \geq 1-\mathsf{negl}(\lambda)\,.$$

**Definition 13** (Unique Provability)**.** For any public key $\mathsf{pk}$ and any input $m$, there does not exist two distinct evaluations which both have accepting proofs, except with negligible probability. Formally, for all adversaries $\mathcal{A}$, which may be computationally unbounded (with at most polynomially many public coin queries),

$$\mathsf{Pr}\left[\begin{array}{c}\mathsf{Verify}_{\mathsf{pk}}(m,h_1,\pi_1)=1\ \wedge\\ \mathsf{Verify}_{\mathsf{pk}}(m,h_2,\pi_2)=1\ \wedge\ h_1\neq h_2\end{array}\;\middle|\;\begin{array}{c}\mathsf{pp}\leftarrow\mathsf{SetUp}(1^\lambda)\\ (\mathsf{pk},m,h_1,\pi_1,h_2,\pi_2)\leftarrow\mathcal{A}(1^\lambda,\mathsf{pp})\end{array}\right]$$

is negligible in the security parameter $\lambda$.

**Definition 14** (Residual Pseudorandomness)**.** Let $\mathcal{A}=(\mathcal{A}_1,\mathcal{A}_2)$ be a PPT adversary and $H$ be a random oracle in the following game:

1. $\mathsf{pp}\leftarrow\mathsf{SetUp}(1^\lambda)$
2. $(\mathsf{sk},\mathsf{pk})\leftarrow\mathsf{KeyGen}(\mathsf{pp})$
3. $(m^*,st)\leftarrow\mathcal{A}_1^{\mathsf{Eval}_{\mathsf{sk}}(\cdot),H(\cdot)}(\mathsf{pk})$
4. $(h_0,\pi_0)\leftarrow\mathsf{Eval}_{\mathsf{sk}}(m^*)$

5. $h_1 \leftarrow \{0,1\}^{2\lambda}$
6. $b \leftarrow \{0,1\}$
7. $b' \leftarrow \mathcal{A}_2^{\mathsf{Eval}_{\mathsf{sk}}(\cdot),H(\cdot)}(\mathsf{pk}, h_b, st)$

$\mathcal{A}$ wins, if $b = b'$ and if it hasn't queried $\mathsf{Eval}_{\mathsf{sk}}(m^*)$. A VRF satisfies the *residual pseudorandomness* property, if $\Pr[\mathcal{A} \text{ wins}] \leq \frac{1}{2} + \mathsf{negl}(\lambda)$.

In this work, we also consider the notion we call *weak unique provability*, also called *computational uniqueness* in [GOT19, EKS+21]. In some cases it is impractical to ensure that the underlying key generation function in a VRF is injective. If this is not the case, an unbounded adversary for Definition 13 can search for a collision $\mathsf{sk}, \mathsf{sk}'$ such that $\mathsf{Verify}_{\mathsf{pk}}(\mathsf{Eval}_{\mathsf{sk}}(m)) = \mathsf{Verify}_{\mathsf{pk}}(\mathsf{Eval}_{\mathsf{sk}'}(m))$, where in general it is not the case that $\mathsf{Eval}_{\mathsf{sk}}(m) = \mathsf{Eval}_{\mathsf{sk}'}(m)$. This relaxed definition accounts for the non-uniqueness of secret keys corresponding to a fixed public key. Instead, it guarantees that colliding secret keys are infeasible to find by requiring a computationally bounded adversary.

**Definition 15** (Weak Unique Provability)**.** It is computationally infeasible to construct a public key $\mathsf{pk}$ and input $m$ such that there exists two distinct evaluations which both have accepting proofs. Formally, for all PPT adversaries $\mathcal{A}$, the probability

$$\Pr\left[\begin{array}{c|c} \mathsf{Verify}_{\mathsf{pk}}(m, h_1, \pi_1) = 1 \ \wedge & \mathsf{pp} \leftarrow \mathsf{SetUp}(1^\lambda) \\ \mathsf{Verify}_{\mathsf{pk}}(m, h_2, \pi_2) = 1 \ \wedge \ h_1 \neq h_2 & (\mathsf{pk}, m, h_1, \pi_1, h_2, \pi_2) \leftarrow \mathcal{A}(1^\lambda, \mathsf{pp}) \end{array}\right]$$

is negligible in the security parameter $\lambda$.

## 5.2 A generic VRF construction

We will use the following syntax throughout this section.

**Definition 16** (Unpredictable function)**.** Let $\mathcal{E}$ and $\mathcal{K}$ be sets, and let

$$\mathcal{W} : \mathcal{E} \times \mathcal{K} \to \mathcal{E}$$

be a deterministic function. We further define $\mathsf{SetUpUF}(1^\lambda)$ as a function which on input a security parameter $\lambda$ outputs an unpredictable function $(\mathcal{W}, \mathcal{E}, \mathcal{K})$ together with an element $E_0 \in \mathcal{E}$ called the *starting element*. We call $(\mathcal{W}, \mathcal{E}, \mathcal{K})$ *unpredictable*, if the one-more evaluation problem on $\mathcal{W}$ is hard.

**Problem 3** (One-more evaluation problem)**.** *Let $\mathcal{O}_k(\cdot)$ be an evaluation oracle, which for given public parameters $\mathsf{pp}$ and on input $m \in \mathcal{K}$ returns $E \leftarrow \mathcal{W}(\mathcal{W}(E_0, m), k)$. Finally, let $\mathcal{A}$ be a PPT adversary for which we define the following game. On input $\lambda$:*

1. $\mathsf{pp} \leftarrow \mathsf{SetUpUF}(1^\lambda)$. *Parse* $\mathsf{pp}$ *as* $(\mathcal{W}, \mathcal{E}, \mathcal{K}, E_0)$.
2. $k \leftarrow\!\!\$\ \mathcal{K}$
3. $E_k \leftarrow \mathcal{W}(E_0, k)$
4. $(m^*, E^*) \leftarrow \mathcal{A}^{\mathcal{O}_k(\cdot)}(E_k, \mathsf{pp})$
5. $\mathcal{A}$ *wins if* $E^* = \mathcal{W}(\mathcal{W}(E_0, m^*), k)$ *and* $m^*$ *has not been queried to* $\mathcal{O}_k(\cdot)$ *before, and loses otherwise.*

Note that the assumption that Problem 3 is hard implies that unpredictable functions $\mathcal{W}$ are collision-resistant, since finding $m, m^*$ where $\mathcal{W}(E_0, m) = \mathcal{W}(E_0, m^*)$ would lead to an adversary who can find one more evaluation in Problem 3. Furthermore, we have that $\mathcal{W}$ is non-commutative in the following sense

$$\mathcal{W}(\mathcal{W}(E_0, m), k) \neq \mathcal{W}(\mathcal{W}(E_0, k), m)\,.$$

Otherwise, one could trivially break the assumption by sampling $m^*$ and computing $\mathcal{W}(E_k, m^*)$.

**Proof system.** We let $\pi \leftarrow \mathsf{NIZK}.P(k, (E_1, E_2, F_1, F_2); \mathsf{pp})$ designate a non-interactive zero-knowledge proof, which for a given input value $k$ and tuples $(E_1, E_2)$ and $(F_1, F_2)$, outputs a proof $\pi$ for the following relation.

$$\mathcal{R} = \left\{ (E_1, E_2), (F_1, F_2), k \;:\; E_2 = \mathcal{W}(E_1, k) \wedge F_2 = \mathcal{W}(F_1, k) \right\}. \tag{5.1}$$

A verifier can then run $0/1 \leftarrow \mathsf{NIZK}.V(\pi, (E_1, E_2, F_1, F_2); \mathsf{pp})$ in order to verify a proof $\pi$ with regards to two tuples $(E_1, E_2)$ and $(F_1, F_2)$. The verifier outputs 0, if it rejects the proof, and 1 otherwise.

### 5.2.1 Constructing VRFs from Unpredictable Functions.

In Figure 5.1, we present our construction for a VRF based on an unpredictable function $\mathcal{W}$ and the proof system $\mathsf{NIZK}$. The VRF evaluation simply consists of two consecutive evaluations of $\mathcal{W}$, first with input $m$, then with input $k$, and a proof that the second path was computed correctly, i.e. using the secret key $k$ that defines the public key $E_k$.

$$E_0 \xrightarrow{\ m\ } E_m = \mathcal{W}(E_0, m) \xrightarrow{\ k\ } E = \mathcal{W}(E_m, k)$$

$$\text{and,} \qquad E_0 \xrightarrow{\ k\ } E_k = \mathcal{W}(E_0, k)$$

At the end, the output, together with the input and the public key, are hashed using a random oracle $H : \{0,1\}^* \to \{0,1\}^{2\lambda}$. This allows to base our underlying hardness assumption on the computational hardness of Problem 3, rather than a decisional one.

$\mathsf{SetUp}(1^\lambda)$

1 : $(\mathcal{W}, \mathcal{E}, \mathcal{K}, E_0) \leftarrow \mathsf{SetUpUF}(1^\lambda)$.
2 : **return** $\mathsf{pp} := (\mathcal{W}, \mathcal{E}, \mathcal{K}, E_0)$

$\mathsf{KeyGen}(\mathsf{pp})$

1 : Parse $\mathsf{pp}$ as $(\mathcal{W}, \mathcal{E}, \mathcal{K}, E_0)$.
2 : $k \leftarrow\!\!\$\ \mathcal{K}$
3 : $E_k := \mathcal{W}(E_0, k)$
4 : **return** $(\mathsf{sk}, \mathsf{pk}) := (k, E_k)$

$\mathsf{Eval}_{\mathsf{sk}}(m; \mathsf{pp})$

1 : Parse $\mathsf{pp}$ as $(\mathcal{W}, \mathcal{E}, \mathcal{K}, E_0)$.
2 : **assert** $m \in \mathcal{K}$
3 : $E_m := \mathcal{W}(E_0, m)$
4 : $E := \mathcal{W}(E_m, \mathsf{sk})$
5 : $\pi_1 \leftarrow \mathsf{NIZK}.P(\mathsf{sk}, (E_0, E_k, E_m, E); \mathsf{pp})$
6 : **return** $h := H(E_k, m, E), \pi := (\pi_1, E)$

$\mathsf{Verify}_{\mathsf{pk}}(h, (\pi_1, E), m; \mathsf{pp})$

1 : Parse $\mathsf{pp}$ as $(\mathcal{W}, \mathcal{E}, \mathcal{K}, E_0)$.
2 : $E_m := \mathcal{W}(E_0, m)$
3 : $b_1 \leftarrow h \stackrel{?}{=} H(E_k, m, E)$
4 : $b_2 \leftarrow \mathsf{NIZK}.V(\pi_1, (E_0, E_k, E_m, E); \mathsf{pp})$
5 : **return** $b_1 \wedge b_2$

Figure 5.1: Verifiable pseudorandom function from unpredictable function $\mathcal{W}$ with setup SetUpUF and zero-knowledge proof NIZK.

*Remark* 5. We note that lifting to a VRF via unpredictable functions is considered folklore in the literature, but its security was not given formal treatment (in the random oracle model in particular) until recently. During the writing of this work, a concurrent result [GS24] also proved the security of this construction in the random oracle model, using a closely related notion of *Verifiable Unpredictable Functions* (VUFs). VUFs are unpredictable functions with the verifiability constraint included in their definition. Given that unpredictable functions equipped with an NIZKPoK for Relation 5.1 satisfy the property of VUF, their result implies that our resulting VRF construction, following the compiler from [GS24, Fig. 9], also satisfies the stronger security notion of *unbiasability*. This property required to guarantee fairness in VRF-based leader election, as required in VRF-based proof-of-stake protocols. We refer the reader to [GS24] for more details.

**Theorem 7** (Provability)**.** *The protocol from Figure 5.1 is provable if the proof system* NIZK *is correct and sound.*

*Proof.* If KeyGen and $\mathsf{Eval}_{\mathsf{sk}}$ are correctly executed, then indeed $h = H(m, \pi_2)$, fulfilling the first verification condition. The second condition immediately follows from the correctness and soundness of NIZK. □

**Theorem 8** (Unique Provability)**.** *The protocol from Figure 5.1 is uniquely provable if* NIZK *is sound and* $\mathcal{W}(E_0, \cdot)$ *is injective, and weak uniquely provable if* $\mathcal{W}(E_0, \cdot)$ *is collision resistant.*

*Proof.* Suppose an adversary's two outputs $(h, (\pi, E))$ and $(h', (\pi', E'))$ are accepting for the same input $m$ and public key $\mathsf{pk}$. We will show that if NIZK is sound and $\mathcal{W}(E_0, \cdot)$ is injective (resp. collision resistant), then with overwhelming probability, $h = h'$. Let $\mathcal{L}$ be the language of Relation 5.1.

Since the proofs are accepting and since NIZK is sound, it must be the case that both

$$h = H(E_k, m, E) \ \wedge \ (E_0, E_k, E_m, E) \in \mathcal{L}\,,$$
$$h' = H(E_k, m, E') \ \wedge \ (E_0, E_k, E_m, E') \in \mathcal{L}\,,$$

up to negligible probability. This implies that both

$$\exists \mathsf{sk} : E_k = \mathcal{W}(E_0, \mathsf{sk}) \ \wedge \ E = \mathcal{W}(E_m, \mathsf{sk})\,,$$
$$\exists \mathsf{sk}' : E_k = \mathcal{W}(E_0, \mathsf{sk}') \ \wedge \ E' = \mathcal{W}(E_m, \mathsf{sk}')\,.$$

In the case where $\mathcal{W}(E_0, \cdot)$ is injective then it cannot be that $\mathsf{sk} \neq \mathsf{sk}'$. If $\mathcal{W}(E_0, \cdot)$ is collision resistant and $\mathsf{sk} \neq \mathsf{sk}'$, then the adversary has found a collision, which occurs with most negligible probability. If $\mathsf{sk} = \mathsf{sk}'$, then $E = E'$ and finally $h = h'$. □

**Theorem 9** (Residual Pseudorandomness)**.** *Let* $\mathcal{A}_{RPR}$ *be an adversary against the residual pseudorandomness game from Definition 14. Let further* $\mathcal{B}_{ZK}$ *be an adversary against the zero-knowledge property of* NIZK *and* $\mathcal{B}_{OME}$ *an adversary against the one-more evaluation problem defined in Problem 3. If* $\mathcal{A}$ *is allowed up to* $q$ *queries to the random oracle* $H$ *and* $n$ *queries to the* $\mathsf{Eval}_{\mathsf{sk}}$ *oracle, then*

$$\mathsf{Adv}(\mathcal{A}_{RPR}) \leq n\mathsf{Adv}(\mathcal{B}_{ZK}) + q\mathsf{Adv}(\mathcal{B}_{OME})$$

*Proof.* We prove the statement using four game hops. Let $\mathsf{Game}_0$ be the residual pseudorandomness game from Definition 14 and let $\mathcal{A}$ be an adversary against $\mathsf{Game}_0$ with advantage $\mathsf{Adv}(\mathcal{A}_{RPR})$. We define an adversary $\mathcal{B}_{ZK}$ against the zero-knowledge property of NIZK and an adversary $\mathcal{B}_{OME}$ against the one-more evaluation problem defined in Problem 3. Let $E_0$ be a publicly available starting curve and let $(\mathsf{sk}, \mathsf{pk}) = (k, E_k)$ be the parameters of the one-more evaluation problem instance.

$\mathsf{Game}_1$**:** Let $\mathcal{S}_{\mathsf{NIZK}}$ be the zero-knowledge simulator of NIZK, which on input $x$ simulates a proof that $x$ is a valid instance of Relation 5.1. We define $\mathsf{Game}_1$ similar to $\mathsf{Game}_0$, except that whenever $\mathcal{A}$ queries $\mathsf{Eval}_{\mathsf{sk}}$ on some input $m$, the game proceeds as follows.

1. Query $(h_0, (\pi_0, E)) \leftarrow \mathsf{Eval}_{\mathsf{sk}}(m)$,
2. compute $E_m = \mathcal{W}(E_0, m)$,
3. query $\pi_0' \leftarrow \mathcal{S}_{\mathsf{NIZK}}(E_0, E_k, E_m, E)$,
4. return $(h_0, (\pi_0', E))$ to $\mathcal{A}$.

If $\mathcal{A}$ can distinguish $\mathsf{Game}_1$ from $\mathsf{Game}_0$, then clearly $\mathcal{A}$ can be used to break the zero-knowledge property of NIZK. Assuming the number of $\mathcal{A}$'s queries to $\mathsf{Eval}_{\mathsf{sk}}$ is bounded by a parameter $n$, then the advantage of adversary $\mathcal{B}_{ZK}^{\mathcal{A}}$ against the zero-knowledge property of NIZK,

$$\mathsf{Adv}(\mathcal{B}_{ZK}) \geq \frac{1}{n}\left|\Pr(\mathcal{A} \text{ wins } \mathsf{Game}_1) - \Pr(\mathcal{A} \text{ wins } \mathsf{Game}_0)\right|.$$

**$\mathsf{Game}_2$:** In this game, whenever $\mathcal{A}$ queries $\mathsf{Eval}_{\mathsf{sk}}$ on some input $m$, the evaluation oracle proceeds as follows:

1. Query $E \leftarrow \mathcal{W}(\mathcal{W}(E_0, m), \mathsf{sk})$,
2. query $h \leftarrow H(E_k, m, E)$,
3. query $\pi_0' \leftarrow \mathcal{S}_{\mathsf{NIZK}}(E_0, E_k, E_m, E)$,
4. return $(h_0, (\pi_0', E))$ to $\mathcal{A}$.

Since $E = \mathcal{W}(\mathcal{W}(E_0, m), k)$, the output $(h_0, (\pi_0', E))$ is equal to $\mathsf{Game}_1$ and therefore $\mathsf{Game}_2$ is perfectly indistinguishable from $\mathsf{Game}_1$.

**$\mathsf{Game}_3$:** This game proceeds exactly as $\mathsf{Game}_2$, except that the game simulates the random oracle, i.e. whenever $\mathcal{A}$ queries the random oracle $H$, instead the game samples $h_0' \leftarrow \{0,1\}^{2\lambda}$ and returns it to $\mathcal{A}$. The game keeps track of $\mathcal{A}$'s queries and returns the same output value for the same input value. Since the game simulates the random oracle perfectly, this game is perfectly indistinguishable from $\mathsf{Game}_2$.

**$\mathsf{Game}_4$:** We define $\mathsf{Game}_4$ in the same way as $\mathsf{Game}_3$, with the exception that $h_0$ is set to be a uniformly sampled string in $\{0,1\}^{2\lambda}$. In particular, it has no relation to the random oracle query on $x^*$. If $H$ is never queried on $x^*$, then $\mathsf{Game}_3$ and $\mathsf{Game}_4$ are perfectly indistinguishable (since $h_i$ is identically distributed) and have the same success probability for the $\mathcal{A}$. Let $X^*$ denote the event that $H$ has been queried on $x^*$ and $\neg X^*$ the event that it hasn't. We have

$$\Pr(\mathcal{A} \text{ wins } \mathsf{Game}_3 \mid \neg X^*) = \Pr(\mathcal{A} \text{ wins } \mathsf{Game}_4 \mid \neg X^*). \tag{5.2}$$

In particular, this allows us to bound the difference in probability to solve either game as follows. Using the law of total probability, we find

$$\begin{aligned}&\left|\Pr(\mathcal{A} \text{ wins } \mathsf{Game}_3) - \Pr(\mathcal{A} \text{ wins } \mathsf{Game}_4)\right| \\ &\quad = \left|(\Pr(\mathcal{A} \text{ wins } \mathsf{Game}_3 \mid X^*) - \Pr(\mathcal{A} \text{ wins } \mathsf{Game}_4 \mid X^*))\Pr(X^*) + \right. \\ &\qquad \left.(\Pr(\mathcal{A} \text{ wins } \mathsf{Game}_3 \mid \neg X^*) - \Pr(\mathcal{A} \text{ wins } \mathsf{Game}_4 \mid \neg X^*))\Pr(\neg X^*)\right|\end{aligned}$$

Using Equation (5.2), the second term vanishes and we can bound

$$\left|\Pr(\mathcal{A} \text{ wins } \mathsf{Game}_3) - \Pr(\mathcal{A} \text{ wins } \mathsf{Game}_4)\right| \leq \Pr(X^*)\,.$$

In a game where $\mathcal{A}$ does make the critical query $x^*$ to the random oracle, the only way it can win $\mathsf{Game}_4$ is by guessing the correct output, since $h_0$ is unrelated to $H(x^*)$, which implies that $\Pr(\mathcal{A} \text{ wins } \mathsf{Game}_4) = \frac{1}{2}$, thus

$$\left|\Pr(\mathcal{A} \text{ wins } \mathsf{Game}_3) - \tfrac{1}{2}\right| \leq \Pr(X^*)\,.$$

**The reduction.** We now show how $\mathcal{B}_{OME}$ can turn an adversary $\mathcal{A}$ against $\mathsf{Game}_4$ into an adversary against the one-more evaluation problem, which will we use to bound $\Pr(X^*)$. Using the reasoning from above, $\mathcal{A}$ must query the random oracle on $(E_k, m^*, E^*)$ in order to successfully distinguish the challenged strings $h_0$ and $h_1$. We therefore only need to consider the case where $\mathcal{A}$ does indeed send $(E_k, m^*, E^*)$ as a query to the random oracle. We argue that submitting this query allows the algorithm $\mathcal{B}_{OME}$ to learn $E^* = \mathcal{W}(\mathcal{W}(E_0, m^*), k)$ and therefore break the one-more evaluation problem. Let the number of random oracle queries by $\mathcal{A}$ be at most $q$. $\mathcal{B}_{OME}$ proceeds as follows. (For simplicity, set $\mathcal{B} = \mathcal{B}_{OME}$ and $\mathcal{A} = \mathcal{A}_{RPR}$.)

1. $\mathcal{B}$ samples $i^* \leftarrow \{1, \ldots, q\}$ and $h_0 \leftarrow \{0,1\}^{2\lambda}$.
2. $\mathcal{B}$ gets as input the public key $E_k = \mathcal{W}(E_0, k)$ and sends it to $\mathcal{A}$.
3. Whenever $\mathcal{A}$ sends its $i$-th query $x_i$ to the random oracle,
   - if $i \neq i^*$ then $\mathcal{B}$ simulates the random oracle truthfully and keeps a list of $\mathcal{A}$'s queries and the related outputs,
   - if $i = i^*$ and $x_{i^*}$ has been queried before, then $\mathcal{B}$ aborts, otherwise it sends $h_0$ to $\mathcal{A}$ and adds $(x_{i^*}, h_0)$ to the random oracle list.
4. Whenever $\mathcal{A}$ sends a query $m$ to $\mathsf{Eval}_k(\cdot)$, $\mathcal{B}$ proceeds as follows:
   (a) query $E \leftarrow \mathcal{O}_k(m)$,
   (b) query $h \leftarrow H(E_k, m, E)$,

(c) use the zero-knowledge simulator $\mathcal{S}$ from NIZK to build an accepting proof $\pi_1 \leftarrow \mathcal{S}((E_0, E_k, E_m, E))$,

(d) send $(h, (\pi_1, E))$ to $\mathcal{A}$.

5. When $\mathcal{A}$ outputs $m^*$, $\mathcal{B}$ checks whether $m^*$ has already been sent by $\mathcal{A}$ as a query to $\mathsf{Eval}_k(\cdot)$. In that case, $\mathcal{B}$ aborts and returns $\perp$. Otherwise $\mathcal{B}$ proceeds as follows:

   (a) sample $h_1 \leftarrow \{0,1\}^{2\lambda}$,

   (b) sample $b \leftarrow \{0,1\}$,

   (c) send $h_b$ to $\mathcal{A}$.

6. For any further query to $H(\cdot)$ or $\mathsf{Eval}_k(\cdot)$, $\mathcal{B}$ proceeds as in steps 3 and 4, respectively.

7. At the end, $\mathcal{A}$ outputs $b'$.

8. If one can parse $x_{i^*}$ as $(E_k, m^*, E^*)$, then $\mathcal{B}$ returns $(m^*, E^*)$.

Assuming that the query $x_{i^*}$ was indeed $(E_k, m^*, E^*)$, then this implies that $\mathcal{B}$ outputs a pair $(m^*, E^*)$ that has not been submitted to $\mathcal{O}_k$, thus breaking the one-more evaluation problem. By randomly guessing one out of $q$ queries by $\mathcal{A}$ to the random oracle, we find that $\mathsf{Adv}(\mathcal{B}_{OME}) \geq \frac{1}{q}\Pr(X^*)$.

Finally, we can compute the advantage of $\mathcal{A}$ against the original game $\mathsf{Game}_0$ using the triangle inequality

$$\begin{aligned}
\mathsf{Adv}(\mathcal{A}_{RPR}) &= \left|\Pr(\mathcal{A} \text{ wins } \mathsf{Game}_0) - \tfrac{1}{2}\right| \\
&= \left|\Pr(\mathcal{A} \text{ wins } \mathsf{Game}_0) - \Pr(\mathcal{A} \text{ wins } \mathsf{Game}_4)\right| \\
&\leq \left|\Pr(\mathcal{A} \text{ wins } \mathsf{Game}_0) - \Pr(\mathcal{A} \text{ wins } \mathsf{Game}_1)\right| \\
&\qquad + \left|\Pr(\mathcal{A} \text{ wins } \mathsf{Game}_3) - \Pr(\mathcal{A} \text{ wins } \mathsf{Game}_4)\right| \\
&\leq n\mathsf{Adv}(\mathcal{B}_{ZK}) + q\mathsf{Adv}(\mathcal{B}_{OME})\,.
\end{aligned}$$

□

**Theorem 10.** *Let* NIZK *be a NIZKPoK for relation 5.1 and let $\mathcal{W}$ be an injective unpredictable function. Then Fig. 5.1 is a verifiable pseudorandom function (VRF) in the random oracle model. If $\mathcal{W}$ is not injective (but still collision resistant), then the protocol is a verifiable pseudorandom function (VRF) with weak unique provability.*

*Proof.* Follows directly from Theorems 7 to 9. □

## 5.3 Instantiation from Radical CGL Isogeny Walks

In this section, we discuss why isogenies are well-suited candidates to instantiate our VRF construction from Section 5.2. In particular, we instantiate the unpredictable function $\mathcal{W}$ using the radical isogeny protocol $\mathsf{CGL}$ from Algorithm 1 over supersingular elliptic curves.

### 5.3.1 Instantiating the Unpredictable Function

Formally, we define $\mathsf{SetUpUF}(1^\lambda)$ as returning $(\mathsf{CGL}, \mathcal{E}, \mathcal{K}, E_0)$, where

- $\mathsf{CGL}$ is the function described in Algorithm 1,
- $\mathcal{E}$ defines the set of supersingular elliptic curves over a finite field $\mathbb{F}_{p^2}$ with parameter size defined with respect to the security parameter $\lambda$ (we discuss actual parameters in Section 5.3.2),
- $E_0 \in \mathcal{E}$ is a starting curve, and
- $\mathcal{K}$ is the set of input strings $\{-1, 1\}^e$.

We can then interpret the VRF input $m \in \{-1, 1\}^e$ as a fixed-length, binary string which defines a walk from the starting curve $E_0$ to some curve $E_m$, from which we then start another walk, this time defined by the server's secret key $k \in \{-1, 1\}^e$ towards the final curve $E$. For the proof system, we use the proof of same input $\mathsf{NIZK}^{/\!/}$ described in Section 4.5. The idea is for the server to show that it used its secret key $k$, which connects $E_0$ to the public key $E_k = \mathsf{CGL}(E_0, k)$, also as an input to evaluate the VRF. We get the picture below

$$E_0 \xrightarrow{\ m\ } E_m = \mathsf{CGL}(E_0, m) \xrightarrow{\ k\ } E = \mathsf{CGL}(E_m, k)$$

$$\text{and,} \qquad E_0 \xrightarrow{\ k\ } E_k = \mathsf{CGL}(E_0, k)\,,$$

where the $\mathsf{NIZK}^{/\!/}$ proves that indeed the pairs $(E_m, E)$ and $(E_0, E_k)$ are connected by the same $k$, according to Relation 4.19. In the next section, we discuss the suitability of $\mathsf{CGL}$ as an unpredictable function and motivate the security of our instantiation.

### 5.3.2 Hardness of the One-more Evaluation Problem

We first discuss considerations which need to be taken into account when implementing the $\mathsf{CGL}$ hash function in order to guarantee collision and pre-image resistance, which are necessary to achieve unpredictability. We then motivate the hardness of the one-more evaluation problem and discuss secure parameter sizes and efficiency of our protocol.

**Non-backtracking walks.** We first note that the walks on the isogeny graph must be non-backtracking, otherwise pre-image and collision resistance can be trivially broken. By using the radical isogeny formulas, backtracking is inherently avoided, by design [CDV20]. Each iteration only allows for two choices of outgoing isogenies, never the dual of the prior step.

**Pre-image resistance.** Given that the walks are non-backtracking, for a curve $E_m$ and a valid output curve $E_k$, computing a pre-image $k$ such that $E_k = \mathsf{CGL}(E_m, k)$ corresponds to computing a cyclic $2^e$-isogeny from $E_m$ to $E_k$ in the isogeny graph, which is Problem 1. The best known classical attacks on this problem run in time $O(2^{e/2})$ via claw-finding attacks [JD11] and $\tilde{O}(\sqrt{p})$ via the volcano-finding algorithm of [DG16]. Hence, we require $e \geq 2\lambda$ and $\log p \geq 2\lambda$ to target $\lambda$ bits of classical security. Note that the best known quantum attacks: quantum claw-finding[Tan09, Zha09] and quantum volcano-finding [BJS14], require $O(2^{e/3})$ and $O(p^{1/4})$ steps respectively.

**Collision resistance and endomorphism ring attacks.** The function $\mathsf{CGL}(E_0, \cdot)$ defined over a fixed input length $e$ is not in general injective. Furthermore, in case the attacker has knowledge of the endomorphism ring of the starting curve, they can compute collisions as described in the attacks of [PL17, Section 4.2], by using the KLPT algorithm [KLPT14]. In particular, in order to compute collisions, the attacker computes cycles in the 2-isogeny graph from the starting curve. This attack can be prevented in two ways, either (1) by using a starting curve $E_0$ with unknown endomorphism ring, via a trusted setup ceremony via techniques described in [BCC+23], or (2) by limiting the message space to be short enough such there are no endomorphisms of length $2^{2r}$ for $r \leq e$. Existence of such cycles would admit a collision in the function $\mathsf{CGL}(E_0, \cdot)$, leading to the scenario where $\mathsf{pk} = \mathsf{CGL}(E_0, \mathsf{sk}) = \mathsf{CGL}(E_0, \mathsf{sk}')$ for distinct $\mathsf{sk}, \mathsf{sk}'$, and $\mathsf{Eval}_{sk}(m) \neq \mathsf{Eval}_{sk'}(m)$, violating weak unique provability, and the same collision would also allow an adversary to find distinct $m, m'$ such that $\mathsf{Eval}_{\mathsf{sk}}(m) = \mathsf{Eval}_{\mathsf{sk}}(m')$, breaking unpredictability. In our setting, we opt for trusted setup, as it is unclear how to construct a prime $p$ and curve $E_0$ such that the conditions of (2) are satisfied. We stress that if an efficient testing procedure for (2) is found, a public parameter could consist of a curve, and a representation of its endomorphism ring, so that users may check the parameters meet the conditions and therefore $\mathsf{CGL}(E_0, \cdot)$ is injective, and the underlying VRF will satisfy full unique provability. We leave this as future work.

**Conjectured Hardness of the One-more Evaluation Problem.** Given that the function $\mathsf{CGL}(E_0, \cdot)$ is pre-image and collision resistant, there are no trivial ways to break the one-more evaluation problem via either recovering the secret key from the public key, or finding collisions in the message evaluation. We further justify why the

functions outputs are unpredictable, that is, why for possibly related messages $m, m'$, the outputs $\mathsf{CGL}(\mathsf{CGL}(E_0, m), k)$ and $\mathsf{CGL}(\mathsf{CGL}(E_0, m'), k)$ appear uncorrelated. We begin with a discussion which justifies our findings, followed by experiments to support our claims.

Although the VRF evaluator reuses the same key directing walks on the supersingular isogeny graph, starting at different curves $E_m, E_{m'}$, there is no real algebraic connection between different evaluations of the VRF. At the $(i+1)$th-step of the secret walk, given that the $i$-th curve is $E_i : y^2 = x^3 + A_i x^2 + C_i x$, the direction of the walk dictated by the bits of $k$ is determined by which root of $\mathsf{Re}(\sqrt{C_i})$ is a quadratic residue modulo $p$. This is an arbitrary choice of ordering, which appears algebraically unrelated to the structure of the graph itself, and depends on the underlying curve coefficients at every step. This makes it difficult to correlate different evaluations of the VRF under the same key.

Furthermore, one can quickly convince oneself that $\mathsf{CGL}(\mathsf{CGL}(E_0, m), k) = \mathsf{CGL}(E_0, m \parallel k)$, if $m$ and $k$ are written with the least significant bit first. Consider the tree of walks starting at the curve $E_0$ dictated by $m \parallel k$ for all $m \in \{-1, 1\}^n$, $k \in \{-1, 1\}^e$. Such a tree, rooted at $E_0$, may be viewed as a depth $2e$ subtree (with a missing branch) of covering graph of the 2-supersingular isogeny graph: the 2-adic Bruhat-Tits tree (see [AIL+21, GGLM24] for details of this correspondence). Observe that if a single bit of a message is different, then the walk dictated by the remaining string is in a completely different branch of the tree.

**Experiments** We further motivate this discussion with two experiments, which aim to show that evaluations of $f(m) = \mathsf{CGL}(E_0, m||k)$ on correlated messages leads to uncorrelated outputs. In the first experiment, we look at the output distribution of $f$ for all $2^e$ inputs, and compare it to the output distribution of $\mathsf{CGL}(E_0, r)$ for an equally many uniformly sampled $r \in \{-1, 1\}^{2e}$. This is to confirm that the output distribution of $f$ is not skewed by the choice of a fixed key. We report our findings in Figure 5.2. We did not find any observable bias.

In the second experiment, we investigate messages that differ only at the last bit. Concretely, we perform the following on computationally feasible parameters $(e, p)$. First, we take a random walk of length $4 \log p$ starting from $E_{-1} : y^2 = x^3 + x$ to obtain a curve in the graph to serve as the starting curve $E_0$. We then sample a uniform key $k \leftarrow\$ \{-1, 1\}^e$. For $s$ iterations:

1. $m \leftarrow\$ \{-1, 1\}^e$, and $r \leftarrow\$ \{-1, 1\}^{2e}$. Let $m'$ be the message that differs from $m$ only at the last entry (i.e. the least significant bit).

2. Compute $E = \mathsf{CGL}(E_0, m||k)$ and $E' = \mathsf{CGL}(E_0, m'||k)$ and $E'' = \mathsf{CGL}(E_0, r)$. The idea is that we will compare the distance between $E$ and $E'$ with the distance

between $E$ and $E''$, a truly independent, random walk. If the two evaluations are correlated, then we would expect the distance between $E$ and $E'$ to be smaller than the distance between $E$ and $E''$.

3. Perform a bounded distance Dijkstra's search to compute $d = d(E, E')$ and $d' = d(E, E'')$, where $d$ is the distance in $G_2(p)$. If $d$ or $d'$ are less than some threshold, we store the occurrence.

We report our results in Figure 5.3. Testing for graph distance $d > 10$ quickly becomes infeasible as the search space grows to $3^d$. We report our results for a 13, 17 and 19 bit prime, for $s = 100,000$ messages. Note that testing over 2000 iterations on a 31-bit prime did not yield even a single pair $(E, E')$ or $(E, E'')$ that were $2^r$-isogenous for $r \leq 10$.

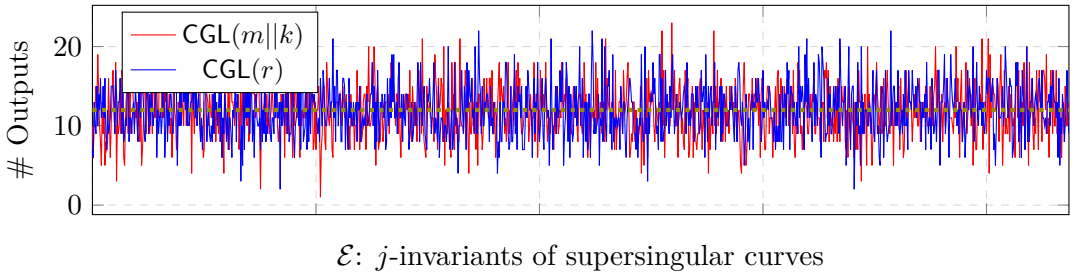



Figure 5.2: Comparison between the distribution of $j$-invariants of $\mathsf{CGL}(m||k)$ and $\mathsf{CGL}(r)$ for a fixed, uniformly sampled $k \in \{-1, 1\}^e$, all possible inputs $m \in \{-1, 1\}^e$, and an equal number of randomly sampled $r \in \{-1, 1\}^{2e}$. The dotted line indicates the uniform distribution on the vertex set. Taken over $p = 2^{17} - 1$, $e = 17$.

**Secure parameters.** In light of the previous discussion, we propose parameters for the VRF in this section. Since generic proofs scale unfavourably with the size of the underlying field of operation; we opt for $\log p \approx 2\lambda$, and trusted setup for obtaining an $E_0$ of unknown endomorphism ring; in the security model where collisions can exist, but should be computationally hard to find.

We opt to set the lengths of the message and key walks to be equal and long enough to ensure hardness of the isogeny problem ($e \approx 2\lambda$).

Given these considerations, we state our security assumption below.

**Conjecture 1** (Radical CGL one-more evaluation problem)**.** *The one-more evaluation problem from Problem 3 is hard, when* $(\mathcal{W}, \mathcal{E}, \mathcal{K}, E_0) \leftarrow \mathsf{SetUp}(1^\lambda)$ *is instantiated as follows.*

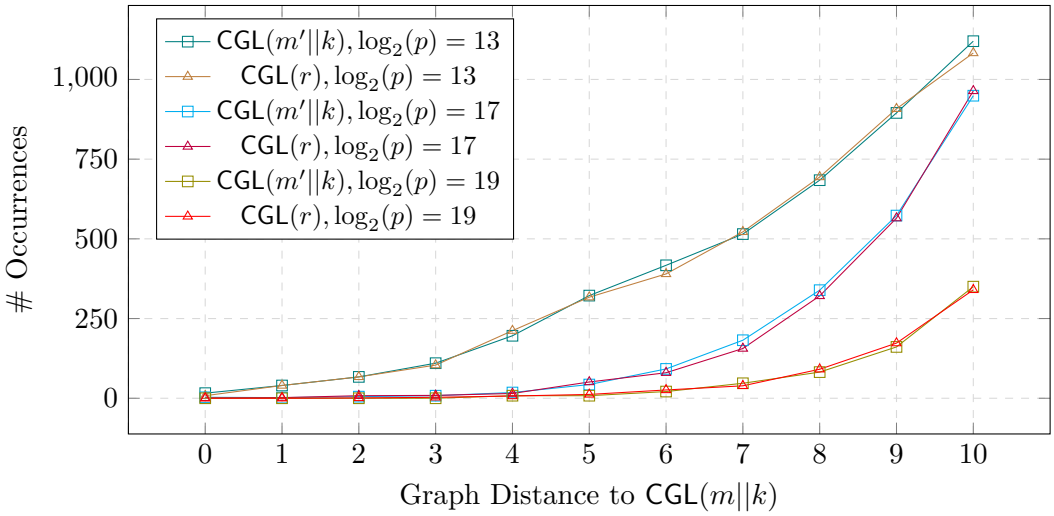



Figure 5.3: Comparison between close occurrences of correlated and random evaluations of the radical isogeny CGL hash function. Correlated messages consist of all $m'$ such that $m'$ differs only at the least significant bit. Experiment was performed using a fixed, randomly sampled key on 10,000 iterations of $m$ over $p \in \{2^{13} - 1, 2^{17} - 1, 2^{19} - 1\}$, $e = \lceil \log_2(p) \rceil$, $k, m, m' \in \{-1, 1\}^e$ and $r \in \{-1, 1\}^{2e}$.

- *$\mathcal{W}$ is the radical CGL hash function* CGL *from Algorithm 1,*
- *$\mathcal{E}$ is the set of supersingular elliptic curves defined over the finite field $\mathbb{F}_{p^2}$, where* $\log p \approx 2\lambda$*,*
- *$\mathcal{K}$ is the set of binary strings $\{-1, 1\}^e$ of fixed length $e \geq 2\lambda$, and*
- *$E_0 \in \mathcal{E}$ is a starting elliptic curve of unknown endomorphism ring (obtained via trusted setup).*

### 5.3.3 Parameter and Proof System Selection

Due to the restrictions of our proof system from Chapter 4, we want to work with fields of characteristic $p \equiv 3 \pmod 4$ such as, for example, quasi-Mersenne primes of the form $p = c2^e - 1$. See the table of proposed parameters for NIST level I security in Table 5.1. We choose our prime in order for their representations to require the minimal number of 32-bit words (i.e. $\log p$ fits in the minimal multiple of 32).

We require post-quantum generic proof systems for R1CS that support $\mathbb{F}_p$-arithmetic with $p \equiv 3 \pmod 4$. This implies that $\mathbb{F}_p^\times$ has two-adicity of one, so we cannot use

| $p$ | $n$ | $\lceil \log p \rceil$ |
|---|---|---|
| $5 \cdot 2^{248} - 1$ | 256 | 251 |

Table 5.1: Proposed parameters for the CGL VRF for NIST level I security ($\lambda = 128$).

Reed-Solomon based SNARKs (such as those based on FRI [BBHR18]), which require two-adicity logarithmic in the size of the circuit. Fortunately, recent developments in SNARK literature have introduced *field agnostic* SNARKs for the context of proving evaluations of the ECDSA verification circuit, which operate over $p = 3 \pmod 4$.

Brakedown [GLS+23], its improvements and variants [Hab23, BFK+24]; BaseFold [ZCF24], and Orion [XZS22] fit our constraints and are viable candidates to instantiate our NIZK. We choose the protocol of [BFK+24] as it is the current state-of-the-art by all performance metrics.

Unfortunately, none of these sources have made a working implementation available which both allows for arbitrary primes and zero-knowledge; so we must rely on the benchmarked costs of the 256-bit prime used in [BFK+24]. Note that their implementation (and indeed, all of the listed candidate SNARKs) do not provide zero-knowledge. We emphasise this is achieved using standard techniques. See Remark 4. We cannot provide a concrete estimate, but these measures are unlikely to cause substantial overhead (and indeed are likely to be accounted for due to the optimisations not implemented in [BFK+24]).

For path lengths of $e = 256$, we end up with 6400 constraints, and hence base our results on circuits of size $2^{13}$. We find the costs outlined in Table 5.2.

| | $T_{\text{KeyGen}}$ | $T_{\text{Eval}}$ | $T_{\text{Verify}}$ | \|PK\| | \|Output\| | Assumption |
|---|---|---|---|---|---|---|
| SL-VRF [BDE+22] | 0.3 ms | 765 ms | 475 ms | 48 B | 40 kB | Hash (LowMC) |
| LaV [ESLR23] | ? | ? | ? | 9 kB | 10 kB | Lattice (MSIS/MLWR) |
| DeuringVRF [Ler25] | 127 ms | 174 ms | 20 ms | 192 B | 256 B | Isogeny (OMIP) |
| Ours (Sec. 5.2) | 5 ms | 85 ms | 30 ms | 64 B | 430 kB | Isogeny (Problem 3) |

Table 5.2: Comparison of our CGL-based VRF from Section 5.2 with other post-quantum VRFs. We compare times for key generation, evaluation and verification of the VRFs, as well as public key and total output size. We estimate performance of our parameters using the reported results of [BFK+24] for a circuit of size $2^{13}$ and a 256-bit prime, and estimating a 256-bit input CGL evaluation as 5 ms.

## 5.4 Instantiation from an SIDH-type Approach

In this section we instantiate the VRF construction from Section 5.2 via an SIDH-type approach, avoiding the use of SNARKs. The relevant notation and details for how this is done are given in Section 2.3.2. We then introduce a sigma protocol, a modified variant of the protocol from [DDGZ22], which requires a decisional assumption. We remark that the constructions resembles SIDH only in the existence of SIDH-squares, which enables the use of the related sigma protocol, hence the isogeny walks are computed via evaluating Vélu's formulae to compute $N$-isogenies for prime power $N$. As per our discussion in the following sections, the attacks of [MMP+23, CD23, Rob23] do not apply to the sigma protocol or the unpredictable function itself, since an attacker does not learn unmasked images of a coprime torsion basis. We start by introducing notation that we will use throughout this section in order to simplify the protocol descriptions.

*Remark* 6. While our overall focus is on achieving a practical SNARK-based approach, we include the construction in this section for two reasons. First, it provides a comparison to the other instantiation, based on a more conventional approach for isogenies via sigma protocols. Second, it is technically more ready for implementation, since it is not reliant on a suitable proof system. However, we do not claim this construction to be more efficient or robust.

**Notation.** Since torsion groups are generated by two independent points, we will be working with elements in a 2-dimensional vector space spanned by the generators. To simplify notation, we will therefore generally use 2-dimensional vectors and matrices. From now on, in this section, we will use the following notation.

Vectors are in bold, e.g. $\mathbf{B} = (P\ Q)^\top$ or $\mathbf{a} = (a_1\ a_2)^\top$, where uppercase letters usually stand for elliptic curve points and lower case for scalars. From this, we can express linear combinations of points simply using the inner dot product

$$\mathbf{a} \cdot \mathbf{B} = a_1 P + a_2 Q.$$

We also denote $\langle \mathbf{B} \rangle = \langle P, Q \rangle$. For an isogeny $\phi$ that pushes through points $\mathbf{B} = (P\ Q)^\top$, we write

$$\phi(\mathbf{B}) = \begin{pmatrix} \phi(P) \\ \phi(Q) \end{pmatrix}.$$

Matrices usually consist of scalars and will be applied to point vectors, e.g.

$$M\mathbf{B} = \begin{pmatrix} \alpha & \beta \\ \gamma & \delta \end{pmatrix} \begin{pmatrix} P \\ Q \end{pmatrix} = \begin{pmatrix} \alpha P + \beta Q \\ \gamma P + \delta Q \end{pmatrix}.$$

### 5.4.1 Instantiating the Unpredictable Function

$\mathcal{E}$ the set of supersingular elliptic curves over $\mathbb{F}_{p^2}$. We assume that all $E \in \mathcal{E}$ have rational $N$-torsion which allows us to compute isogenies of degree $N$ without working in field extensions. We work with elliptic curves of Montgomery form (with second coefficient $B = 1$) such that for all $E \in \mathcal{E}$, the curve is represented by the equation

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

We present our unpredictable function in Algorithm 2 below. We fix the key and message spaces to be fixed length elements of $\mathcal{K} = \mathbb{Z}_N$. We choose inputs of this form, rather than $\mathbb{P}^1(\mathbb{Z}_N)$, in order avoid the possibility of distinct keys (or messages) leading to an equivalent isogeny. The basic idea is that each element $m \in \mathcal{K}$ defines a unique kernel of an isogeny. To this end, we need to define a basis $\mathbf{B}$ of $E_0[N]$, so that we can compute an isogeny given by the kernel $\langle (1, m)^\top \cdot \mathbf{B} \rangle$. In order to ensure a deterministic output that is uniquely defined by the input $m$, we also need this basis to be deterministic if we want to handle arbitrary evaluations of the walk function. We define $\mathcal{B}_N$ as the deterministic $N$-torsion basis sampling algorithm, which on input an elliptic curve $E$ outputs a basis $\mathbf{B}$ that generates $E[N]$. We write this as

$$\mathbf{B} \leftarrow \mathcal{B}_N(E)\,.$$

For the rest of this section, we assume $N = 2^e$ for some $e \in \mathbb{N}$. Deterministic bases can for example be sampled as in the technique described in [ZSP+18, Algorithm 3.1]. To explicitly avoid backtracking, we can further employ the technique from [DPB24, Algorithm 1] that transforms any basis into a non-backtracking basis. This is also important for the second step of the protocol, where the evaluator needs to compute the secret isogeny from the curve $E_m$. In order to prevent backtracking, along with the message $m$, our unpredictable function parses $(E, T)$ as public input: the starting curve $E$, and a kernel generator $T$ for a 2-isogeny which corresponds to the backtracking direction of the previous isogeny walk. This ensures that VRF evaluations are inherently non-backtracking. For simplicity, we write $\mathsf{CGL\text{-}Vélu}(E, m)$ when the backtracking check is not required, such as when the initial walk of the VRF evaluation takes place.

This construction is a variant of the CGL hash function [CLG09] as first described in [DPB24]. In [DPB24, Section 3] it was proven that this defines a one-way function given the hardness of Problem 1. Note that our algorithm outputs an elliptic curve and not a $j$-invariant.

**Algorithm 2** CGL-Vélu$((E_0, T_0), m)$: Variant of the CGL Hash function from [DPB24]

**Require:** Supersingular elliptic curve $E$, (optional) point $T \in E[2]$, message $\mathbf{m} \in \mathcal{K}$
**Ensure:** Evaluation $E'$, $T'$

$\mathbf{B} \leftarrow \mathcal{B}_N(E)$
**if not** $T_0 \stackrel{?}{=} NULL$ **then** ▷ Check that $K = \langle \mathbf{m} \cdot \mathbf{B} \rangle$ does not backtrack on $T$
  Parse $\mathbf{B}$ as $(P, Q)^\top$.
  **if** $[2^{e-1}]P = T$ **then**
    $\mathbf{B} \leftarrow (Q, P)^\top$
  **else if** $[2^{e-1}]Q \neq T$ **then**
    $\mathbf{B} \leftarrow (P, P + Q)^\top$
  **end if**
**end if**
$E' \leftarrow E/\langle \mathbf{m} \cdot \mathbf{B} \rangle$
Obtain the 2-torsion point $T'$ on $E'$ that backtracks on $\phi : E \to E'$.
**return** $E', T'$

### 5.4.2 A Sigma Protocol for Proving Evaluations

Given a deterministic basis sampling algorithm, we define the following relation

$$\mathcal{R}_N = \left\{ (E, F), (E', F'), \mathbf{k} \;\middle|\; \begin{array}{c} F = E/\langle \mathbf{k} \cdot \mathbf{B} \rangle \wedge F' = E'/\langle \mathbf{k} \cdot \mathbf{B}' \rangle \\ \wedge \mathbf{B} \leftarrow \mathcal{B}_N(E) \wedge \mathbf{B}' \leftarrow \mathcal{B}_N(E') \end{array} \right\}, \tag{5.3}$$

and the associated language $\mathcal{L}_N$. The idea of this relation is to show that the same key $\mathbf{k}$ has been used to connect two different pairs of elliptic curves uniquely. Note that $\mathcal{R}_N$ does not uniquely fix the witness $\mathbf{k}$, since $\langle P \rangle = \langle cP \rangle$ for any $c$ coprime to $P$'s order. As a result, given a witness $\mathbf{k}$, any $c\mathbf{k}$ where $\gcd(c, N) = 1$ is also a valid witness for a given statement. This is however not an issue, as equivalent kernels also lead to equivalent isogenies, so that any of the keys from the set, together with a deterministic basis generation algorithm, uniquely fixes the isogeny. This will be enough for our VRF. Throughout this section, one can see the key $\mathbf{k}$ as defined up to scalar multiplication by such a $c$. Nevertheless, in order to minimise key size, key generation should be performed by sampling $k \leftarrow\!\!\$\, \mathbb{Z}_N$ and setting $\mathbf{k} = (1, k)^\top$, (but we consider scalar multiples equivalent).

We show our zero-knowledge proof in Figure 5.4. We require a secure (post-quantum) commitment scheme $\mathsf{Commit} : \{0,1\}^* \to \{0,1\}^{2\lambda}$. This may be instantiated via a hash-based commitments (i.e. $\mathsf{Commit}(x) = H(x||r)$, where $r$ is a random nonce, for a random oracle $H$). Our proof is based on the following pair of commutative diagrams, in which we illustrate the steps of the proof. We note that $[\phi]_*\psi$ denotes the push-through

of $\psi$ through $\phi$ i.e. $\ker([\phi]_*\psi) = \phi(\ker(\psi))$, in order to construct an SIDH square as in Definition 11.

$$\begin{array}{ccc}
E, \mathbf{A} & \xrightarrow{\quad\phi\quad} & F \\
\Big\downarrow{\scriptstyle \psi} & & \Big\downarrow{\scriptstyle \tilde{\psi}=[\phi]_*\psi} \\
E_1, \mathbf{A_1} = M\psi(\mathbf{A}), \mathbf{B_1} & \xrightarrow[\tilde{\phi}=[\psi]_*\phi]{} & E_2, \mathbf{B_2} = \tilde{\phi}(\mathbf{B_1}) \\
E', \mathbf{A}' & \xrightarrow{\quad\phi'\quad} & F' \\
\Big\downarrow{\scriptstyle \psi'} & & \Big\downarrow{\scriptstyle \tilde{\psi}'=[\phi']_*\psi'} \\
E'_1, \mathbf{A'_1} = M\psi'(\mathbf{A}'), \mathbf{B'_1} & \xrightarrow[\tilde{\phi}'=[\psi']_*\phi']{} & E'_2, \mathbf{B'_2} = \tilde{\phi}'(\mathbf{B'_1})
\end{array}$$

In order to simulate protocol transcripts for the following protocol, we introduce Problem 4, a close variant of [Bas24b, Problem 7]. However, this problem is strictly harder than the prior, which includes additional torsion point information that leaks the action of the horizontal isogenies. We remark that our version of double DSSP is unrelated to the double DSSP problem from [DDGZ22, Definition 8], which involves two isogeny squares with a common edge. For improved clarity we write our distributions via sampling algorithms.

**Problem 4** (Double DSSP Problem)**.** *Given supersingular curves $E, F, E', F'$, $N$-torsion bases $\mathbf{A} \in E[N]^2$ and $\mathbf{A}' \in E'[N]^2$ and isogenies $\phi : E \to F$ and $\phi' : E' \to F'$, such that $\ker\phi = \langle \mathbf{k} \cdot \mathbf{A} \rangle$ and $\ker\phi' = \langle \mathbf{k} \cdot \mathbf{A}' \rangle$ for some $\mathbf{k} \in \mathbb{Z}_N^2$, construct a PPT distinguisher $\mathcal{A}$ with non-negligible advantage, which takes as input $(E, F, E', F', \mathbf{A}, \mathbf{A}')$ for the distributions of $\mathcal{D}_0$ and $\mathcal{D}_1$ on input $(E, F, E', F', \mathbf{A}, \mathbf{A}', \mathbf{k})$:*

$\mathcal{D}_0(E, F, E', F', \mathbf{A}, \mathbf{A}', \mathbf{k})$

1 : *Sample random $N'$-isogenies* $\psi : E \to E_1, \psi' : E' \to E'_1$
2 : $M \leftarrow\$ SL_2(\mathbb{Z}_N), \kappa \leftarrow \mathbf{k}M^{-1}$
3 : $\mathbf{A}_1 \leftarrow M\psi(\mathbf{A})$ *and* $\mathbf{A'}_1 \leftarrow M\psi'(\mathbf{A}')$
4 : *Construct $N$-isogenies* $\tilde{\phi} : E_1 \to E_2$ *and* $\tilde{\phi}' : E'_1 \to E'_2$ *such that* $\ker \tilde{\phi} = \kappa \cdot \mathbf{A_1} = \psi(\ker \phi)$ *and* $\ker \tilde{\phi}' = \kappa \cdot \mathbf{A'_1} = \psi'(\ker \phi')$
5 : **return** $(E_1, E_2, \mathbf{A_1}, E'_1, E'_2, \mathbf{A'_1}, \kappa)$

$\mathcal{D}_1(E, F, E', F', \mathbf{A}, \mathbf{A}', \mathbf{k})$

1 : *Sample random $N'$-isogenies* $\psi : E \to E_1, \psi' : E' \to E'_1$
2 : $M \leftarrow\$ SL_2(\mathbb{Z}_N), \kappa \leftarrow\$ \mathbb{Z}_N^2$
3 : $\mathbf{A}_1 \leftarrow M\psi(\mathbf{A})$ *and* $\mathbf{A'}_1 \leftarrow M\psi'(\mathbf{A}')$
4 : *Construct $N$-isogenies* $\tilde{\phi} : E_1 \to E_2$ *and* $\tilde{\phi}' : E'_1 \to E'_2$ *such that* $\ker \tilde{\phi} = \kappa \cdot \mathbf{A_1}$ *and* $\ker \tilde{\phi}' = \kappa \cdot \mathbf{A'_1}$
5 : **return** $(E_1, E_2, \mathbf{A_1}, E'_1, E'_2, \mathbf{A'_1}, \kappa)$

*We note that the only difference between the two distributions is that in $\mathcal{D}_0$ the curves and isogenies are consistent with the two SIDH squares in the diagram above, (where $\kappa = \mathbf{k}M^{-1}$), while in $\mathcal{D}_1$ they are not, where $\tilde{\phi}$ and $\tilde{\phi}'$ are essentially correlated, randomly sampled isogenies. The first distribution is taken over the uniformly at random sampling of $\psi, \psi'$ and $M$, while the second distribution is taken over uniformly at random sampling of $\psi, \psi', \kappa$, and $M$.*

We note that the double DSSP problem requires $\log N' \approx \log N$ to guarantee the hardness of solving for an $N'$-isogeny $\psi : E \to E_1$ (or $\psi' : E' \to E'_1$). Recovering such an isogeny would allow a distinguisher to recover the hidden matrix $M$, and thus "complete the SIDH square" by checking that $\tilde{\phi}$ and $\phi$ are indeed parallel isogenies.

**Theorem 11.** *The protocol in Figure 5.4 is complete, three-special sound and (weak) honest-verifier zero-knowledge for the relation $\mathcal{R}_N$, assuming the hardness of the Double DSSP problem (Problem 4).*

*Proof.*

**Completeness.** The correctness of the isogeny kernels immediately follows from the definition in the proof step. The same argument applies to the point maps. In the verification step for the case $\mathbf{c} = -\mathbf{1}$, we have that

$$(N'M)^{-1}\hat{\psi}(\mathbf{A_1}) = (N'M)^{-1}\hat{\psi}(M\psi(\mathbf{A})) = N'^{-1}M^{-1}M\hat{\psi} \circ \psi(\mathbf{A}) = \mathbf{A}\,,$$

$P(E, F, E', F', \mathbf{k}; pp)$ $V(E, F, E', F'; pp)$

1 : Generate bases $\mathbf{A} \leftarrow \mathcal{B}_N(E)$ and $\mathbf{A}' \leftarrow \mathcal{B}_N(E')$
2 : Compute random $N'$-isogenies $\psi : E \to E_1$,
and $\psi' : E' \to E_1'$
3 : $M \leftarrow\$ \mathrm{SL}(2, \mathbb{Z}_N)$
4 : Set $\mathbf{A_1} \leftarrow M\psi(\mathbf{A}), \mathbf{A_1'} \leftarrow M\psi'(\mathbf{A}')$, and $\kappa \leftarrow \mathbf{k}M^{-1}$
5 : Sample bases $\mathbf{B_1} \leftarrow E_1[N']^2$ and $\mathbf{B_1'} \leftarrow E_1'[N']^2$
6 : Compute $\mathbf{s}, \mathbf{s}'$ such that:
$\langle \mathbf{s} \cdot \mathbf{B_1} \rangle = \ker \hat{\psi}$ and $\langle \mathbf{s}' \cdot \mathbf{B_1'} \rangle = \ker \hat{\psi}'$
7 : Compute $\mathbf{B_2} = \tilde{\phi}(\mathbf{B_1})$ and $\mathbf{B_2'} = \tilde{\phi}'(\mathbf{B_1'})$ where
$\ker \tilde{\phi} = \langle \psi(\mathbf{k} \cdot \mathbf{A}) \rangle$ and $\ker \tilde{\phi}' = \langle \psi'(\mathbf{k} \cdot \mathbf{A}') \rangle$
8 : $r_1, \ldots, r_5 \leftarrow\$ \{0,1\}^\lambda$
9 : $\mathsf{C}_1 \leftarrow \mathsf{Commit}(E_1, \mathbf{A_1}, \mathbf{B_1}, E_1', \mathbf{A_1}', \mathbf{B_1'}; r_1)$
$\mathsf{C}_2 \leftarrow \mathsf{Commit}(E_2, \mathbf{B_2}, E_2', \mathbf{B_2'}; r_2), \mathsf{C}_3 \leftarrow \mathsf{Commit}(\mathbf{s}, \mathbf{s}'; r_3)$
$\mathsf{C}_4 \leftarrow \mathsf{Commit}(M; r_4), \mathsf{C}_5 \leftarrow \mathsf{Commit}(\kappa; r_5)$

$\mathsf{C}_1, \mathsf{C}_2, \mathsf{C}_3, \mathsf{C}_4, \mathsf{C}_5$ →

$c \leftarrow\$ \{-1, 0, 1\}$ : 1

← $c$

**if** $c = -1$ **then**
10 : Send $\mathsf{open}_{-1}$ consisting of the inputs to $\mathsf{C}_1, \mathsf{C}_3, \mathsf{C}_4$
**elseif** $c = 0$ **then**
11 : Send $\mathsf{open}_0$ consisting of the inputs to $\mathsf{C}_1, \mathsf{C}_2, \mathsf{C}_5$
**elseif** $c = 1$ **then**
12 : Send $\mathsf{open}_1$ consisting of the inputs to $\mathsf{C}_2, \mathsf{C}_3$
**fi**

$\mathsf{open}_c$ →

**if** $c = -1$ **then**
Compute $\ker \hat{\psi} = \langle \mathbf{s} \cdot \mathbf{B_1} \rangle$ and $\ker \hat{\psi}' = \langle \mathbf{s}' \cdot \mathbf{B_1'} \rangle$ : 2
Verify $\hat{\psi}$ and $\hat{\psi}'$ are $N'$-isogenies connecting
$E_1 \to E$ and $E_1' \to E'$, respectively : 3
Compute $\mathbf{A} \leftarrow \mathcal{B}_N(E)$ and $\mathbf{A}' \leftarrow \mathcal{B}_N(E')$ : 4
Verify $N'M\mathbf{A} \stackrel{?}{=} \hat{\psi}(\mathbf{A_1})$ and $N'M\mathbf{A}' \stackrel{?}{=} \hat{\psi}'(\mathbf{A_1'})$ : 5
**elseif** $c = 0$ **then**
Compute $\ker \tilde{\phi} = \langle \kappa \cdot \mathbf{A_1} \rangle$ and $\ker \tilde{\phi}' = \langle \kappa \cdot \mathbf{A_1'} \rangle$ : 6
Verify that $\tilde{\phi}$ and $\tilde{\phi}'$ are $N$-isogenies connecting
$E_1 \to E_2$ and $E_1' \to E_2'$, respectively : 7
Verify $\mathbf{B_2} \stackrel{?}{=} \tilde{\phi}(\mathbf{B_1})$ and $\mathbf{B_2'} \stackrel{?}{=} \tilde{\phi}'(\mathbf{B_1'})$ : 8
**elseif** $c = 1$ **then**
Compute $\ker \hat{\tilde{\psi}} \leftarrow \langle \mathbf{s} \cdot \mathbf{B_2} \rangle$ and $\ker \hat{\tilde{\psi}}' \leftarrow \langle \mathbf{s}' \cdot \mathbf{B_2'} \rangle$ : 9
Verify $\hat{\tilde{\psi}}$ and $\hat{\tilde{\psi}}'$ are $N'$-isogenies connecting
$E_2 \to F$ and $E_2' \to F'$, respectively : 10

Figure 5.4: 3-special sound $\Sigma$-protocol for Relation 5.3.

since $\hat{\psi} \circ \psi = \deg \psi = N'$.

**3-Special Soundness.** We show how to extract $\mathbf{k}$ from three accepting transcripts for $c = -1$, $c = 0$ and $c = 1$. Since Commit is computationally binding, the prover is not able to open the commitments to different values with all but negligible probability. We can therefore assume that (omitting the random nonces $r_1, \ldots, r_5$)

$$\begin{aligned}\mathsf{open}_{-1} &= (E_1, \mathbf{A_1}, \mathbf{B_1}, E_1', \mathbf{A_1}', \mathbf{B_1}', \mathbf{s}, \mathbf{s}', M)\\ \mathsf{open}_0 &= (E_1, \mathbf{A_1}, \mathbf{B_1}, E_1', \mathbf{A_1}', \mathbf{B_1}', E_2, \mathbf{B_2}, E_2', \mathbf{B_2}', \kappa)\\ \mathsf{open}_1 &= (E_2, \mathbf{B_2}, E_2', \mathbf{B_2}', \mathbf{s}, \mathbf{s}')\end{aligned}$$

with all values consistent along the three transcripts. We can directly recover $\mathbf{k} = \kappa M$ from $\mathsf{open}_{-1}$ and $\mathsf{open}_0$, but first have to argue that this is indeed the value we want to extract.

The first thing we want to note, is that both sets of elliptic curves $(E, E_1, F, E_2)$ and $(E', E_1', F', E_2')$ each define an SIDH-square. We only show the former as the latter follows in the exact same way. We know that $\ker \hat{\psi} = \langle \mathbf{s} \cdot \mathbf{B_1} \rangle$ and $\ker \hat{\tilde{\psi}} = \langle \mathbf{s} \cdot \mathbf{B_2} \rangle$ and that $\mathbf{B_2} = \tilde{\phi}(\mathbf{B_1})$, therefore

$$\ker \hat{\tilde{\psi}} = \langle \tilde{\phi}(\mathbf{s} \cdot \mathbf{B_1}) \rangle$$

defines the push-forward isogeny of $\hat{\psi}$ through $\tilde{\phi}$ and $(E, E_1, F, E_2)$ is indeed an SIDH square. Now, since $\ker \tilde{\phi} = \langle \kappa \cdot \mathbf{A_1} \rangle$ defines a kernel of degree $N$, [DDGZ22, Lemma 2] implies that there exists an isogeny $\phi : E \to F$ of degree $N$. The kernel of $\phi$ is then the push-forward of $\ker \tilde{\phi}$ through $\hat{\psi}$,

$$\ker \phi = \hat{\psi}(\ker \tilde{\phi}) = \langle \hat{\psi}(\kappa \cdot \mathbf{A_1}) \rangle$$

Plugging in $\mathbf{A_1} = M\psi(\mathbf{A})$, we find

$$\ker \phi = \langle (\kappa \cdot M) \cdot \mathbf{A} \rangle .$$

In exactly the same way, we also find

$$\ker \phi' = \langle (\kappa \cdot M) \cdot \mathbf{A}' \rangle .$$

The extractor can therefore return $\kappa \cdot M$, which corresponds to a valid instance-witness pair for Relation 5.3, since by construction $\mathbf{A} = \mathcal{B}_N(E)$ and $\mathbf{A}' = \mathcal{B}_N(E')$.

**(Weak) Honest-Verifier Zero-Knowledge.** Let $\mathcal{S}$ be a simulator, which on input $(E, F, E', F')$, and challenge $c$, outputs the following, conditioned on $c$:

$\mathbf{c} = -\mathbf{1}$ : $\mathcal{S}$ computes the inputs to $\mathsf{C}_1$, $\mathsf{C}_3$ and $\mathsf{C}_4$ as in the real execution of the protocol. Further, $\mathcal{S}$ samples[2] $\mathsf{C}_2, \mathsf{C}_5 \leftarrow \{0,1\}^{2\lambda}$. Since $\mathsf{C}_1$, $\mathsf{C}_3$ and $\mathsf{C}_4$ are computed as in the real execution of the protocol, it is clear that the protocol accepts and that these values are perfectly indistinguishable from the real execution of the protocol. By the perfect hiding of $\mathsf{Commit}$, $\mathsf{C}_2$ and $\mathsf{C}_5$ are also indistinguishable from the real execution.

$\mathbf{c} = \mathbf{0}$ : $\mathcal{S}$ generates $\mathsf{C}_1$, $\mathsf{C}_3$ and $\mathsf{C}_4$ exactly as in the real execution of the protocol. Then, $\mathcal{S}$ samples $\kappa \leftarrow \mathbb{Z}_N^2$ and $r_5 \leftarrow \{0,1\}^\lambda$ and computes $\mathsf{C}_5 = \mathsf{Commit}(\kappa; r_5)$. Now, $\mathcal{S}$ computes $\tilde{\phi} : E_1 \to E_2$ and $\tilde{\phi}' : E_1' \to E_2'$ given by the kernels $\ker \tilde{\phi} = \langle \kappa \cdot \mathbf{A_1} \rangle$ and $\ker \tilde{\phi}' = \langle \kappa \cdot \mathbf{A_1'} \rangle$, and pushes through $\mathbf{B_2} = \tilde{\phi}(\mathbf{B_1})$ and $\mathbf{B_2'} = \tilde{\phi}'(\mathbf{B_1'})$. Finally $\mathcal{S}$ samples $r_2 \leftarrow \{0,1\}^\lambda$ and computes $\mathsf{C}_2 = \mathsf{Commit}(E_2, \mathbf{B_2}, E_2', \mathbf{B_2'}; r_2)$, which will result in an accepting protocol. We note that the commitment phase is identically distributed by the perfect hiding property of the commitment scheme. What remains is to show that $\mathsf{open}_0$ is computationally indistinguishable from real executions. For this we rely on Problem 4. We show how a distinguisher for real and simulated transcripts yields a distinguisher for the Double DSSP problem (Problem 4). The double DSSP distinguisher constructs the commitments and openings for $\mathsf{C}_3$, $\mathsf{C}_4$, in the same way as the simulator, and it uses the sample from the distribution to build the remaining commitments $\mathsf{C}_1$ and $\mathsf{C}_2$ and $\mathsf{C}_5$. Given sample $(E_1, E_2, \mathbf{A}_1, \tilde{\phi}, E_1', E_2', \mathbf{A}_1', \kappa)$, the distinguisher samples bases $\mathbf{B}_1$ and $\mathbf{B}_1'$ and computes $\mathbf{B}_2 = \tilde{\phi}(\mathbf{B}_1)$ and $\mathbf{B}_2' = \tilde{\phi}'(\mathbf{B_1'})$, where $\tilde{\phi}$ and $\tilde{\phi}'$ are the isogenies given by kernels $\kappa \cdot \mathbf{A}_1$ and $\kappa \cdot \mathbf{A}_1'$ respectively. The distinguisher then computes $\mathsf{C}_1 = \mathsf{Commit}(E_1, \mathbf{A_1}, \mathbf{B_1}, E_1', \mathbf{A_1}', \mathbf{B_1'}; r_1)$, $\mathsf{C}_2 = \mathsf{Commit}(E_2, \mathbf{B_2}, E_2', \mathbf{B_2'}; r_2)$, $\mathsf{C}_3 = \mathsf{Commit}(\mathbf{s}, \mathbf{s}'; r_3)$, $\mathsf{C}_4 = \mathsf{Commit}(M; r_4)$ and $\mathsf{C}_5 = \mathsf{Commit}(\kappa; r_5)$. Finally, the distinguisher sends this transcript to the zero-knowledge distinguisher for $c = 0$. Clearly, the distribution of transcripts is identical to the real protocol executions when the Double DSSP distribution is $\mathcal{D}_0$, and the distributions of transcripts are identical to the simulated protocol when the Double DSSP distribution is $\mathcal{D}_1$.

$\mathbf{c} = \mathbf{1}$ : $\mathcal{S}$ samples two $N'$-isogenies $\tilde{\psi} : F \to E_2$ and $\tilde{\psi}' : F' \to E_2'$, then samples random bases $\mathbf{B_2}$ and $\mathbf{B_2'}$ on $E_2$ and $E_2'$, respectively and solves for $\mathbf{s}, \mathbf{s}'$, such that $\langle \mathbf{s} \cdot \mathbf{B_2} \rangle = \ker \hat{\tilde{\psi}}$ and $\langle \mathbf{s}' \cdot \mathbf{B_2'} \rangle = \ker \hat{\tilde{\psi}}'$. Finally $\mathcal{S}$ computes $\mathsf{C}_2 = \mathsf{Commit}(E_2, \mathbf{B_2}, E_2', \mathbf{B_2'}; r_2)$ and $\mathsf{C}_3 = \mathsf{Commit}(\mathbf{s}, \mathbf{s}'; r_3)$ for $r_2, r_3 \leftarrow \{0,1\}^\lambda$ and samples $\mathsf{C}_1, \mathsf{C}_4, \mathsf{C}_5 \leftarrow \{0,1\}^{2\lambda}$. $E_2$ and $E_2'$ are sampled equivalently to the real protocol, in the sense that there always exist $\psi : E \to E_1$ and

[2] Note that this assumes the commitment scheme has uniform output $\{0,1\}^{2\lambda}$, which differs slightly from the definition of perfect hiding. To precisely match the security definitions, the simulator can just set $\mathsf{C}_2$ and $\mathsf{C}_5$ to be commitments to arbitrary dummy messages.

$\psi' : E' \to E'_1$ to generate a SIDH square, and are therefore indistinguishable. It is clear that the protocols accepts. For every simulated transcript where $\mathbf{B_2}$ and $\mathbf{B'_2}$ are randomly sampled, there is a corresponding real protocol transcript where the random sampling of $\mathbf{B_1}$, $\mathbf{B'_1}$ leads to an identical transcript. Therefore they are distributed identically. This implies that $\mathbf{s}$ and $\mathbf{s}'$ are also indistinguishable. Finally, since $\mathsf{Commit}$ is perfectly hiding, $\mathsf{C}_1$, $\mathsf{C}_4$ and $\mathsf{C}_5$ are also indistinguishable from the real execution.

□

Using the Fiat-Shamir transform, we can turn this proof system in to a non-interactive zero-knowledge proof. Since a malicious prover can cheat with probability $\frac{2}{3}$, we have to repeat the protocol $\lambda \log_2 3$ times to get a negligible soundness error of $2^{-\lambda}$.

We refer to the non-interactive version of the proof as $\mathsf{NIZK}_{\mathcal{L}_N}$ and define the following API.

- The proof $\pi \leftarrow \mathsf{NIZK}_{\mathcal{L}}.P(s; k)$ takes as input the statement $s$ and the witness $\mathbf{k}$ and returns a proof $\pi$.
- The verification $0/1 \leftarrow \mathsf{NIZK}_{\mathcal{L}}.V(s; \pi)$ takes as inputs the statement and a proof, an returns 0 or 1.

**Comparison to other proof systems.** We would like to discuss similarities and differences of our proof system with concurrent proof systems from the isogeny literature. In fact, our proof system builds on the proof of isogeny knowledge introduced in [DDGZ22, Section 5], but extends it in a way to prove knowledge of two isogenies and a specific relation between them, which is defined through the language $\mathcal{L}_N$. In particular, our system evokes a concept similar to the *zero-knowledge proof of equality of appended values* from the OPRF design by Boneh, Kogan and Woo [BKW20, Section 6] and the *proof of parallel isogenies* from the follow-up work by Basso [Bas24b, Section 6]. These proof systems try to convince a verifier that some isogeny has been computed "in the same way" as some previously computed isogeny, the latter usually being the one connecting the starting curve to the public key. In that sense, these proofs show that the secret key has also been used in the new isogeny computation, which convinces the verifier that the prover knows and has used the secret key correctly in its evaluation. This idea is also reflected in our construction, yet the proof systems from [BKW20, Bas24b] have a much stronger requirement, i.e. that the isogenies are *parallel* in the traditional sense, i.e. the four curves involved form a SIDH square. The issue with these kinds of proofs is that the prover needs auxiliary points, which after the SIDH attacks [MMP+23, CD23, Rob23] need to be masked [FMP23]. This results in very large parameters (for example [Bas24b] uses a 8868-bit prime) and therefore expensive computations and large outputs. There doesn't seem to be an easy way

around this when building OPRFs, but our VRF is much more flexible in that regard. As long as the prover shows to have used its secret key correctly and the output is deterministic, there is no need for any notion of parallel isogenies. Our proof therefore only forces the prover to use the same secret key as input, which deterministically fixes the output of the VRF, and in particular prevents the need of using masked points, strongly reducing the necessary parameters. As a further bonus, active attacks in the style of [GPST16, BKM+21], taking advantage of reusing the same point maps multiple times also do not apply to our setting.

### 5.4.3 Hardness of the One-more Evaluation Problem

We introduce the following hardness assumption, which is essentially equivalent to Conjecture 1, but in the context of a different variant of the CGL hash function, which is Algorithm 2 rather than Algorithm 1. We note that in order to use the proof of parallel evaluation from Section 5.4.2, we are now required to work over a prime which supports coprime torsion subgroups $E[N]$ and $E[N']$. In this case, for efficiency and compatibility, we rely on the parameterisation used in SIDH, which sets $N = 2^e$ and $N' = 3^f$, and choosing a prime $p = 2^e 3^f - 1$ such that $2^e \approx 3^f \approx 2^{2\lambda}$ in order to guarantee rationality of torsion.

**Conjecture 2** (Vélu CGL one-more evaluation problem)**.** *The one-more evaluation problem from Problem 3 is hard, when* $(\mathcal{W}, \mathcal{E}, \mathcal{K}, E_0) \leftarrow \mathsf{SetUp}(1^\lambda)$ *is instantiated as follows.*

- $\mathcal{W}$ *is the Vélu CGL hash function* CGL-Vélu *from Algorithm 2.*
- $\mathcal{E}$ *is the set of supersingular elliptic curves defined over the finite field* $\mathbb{F}_{p^2}$*, where* $p = 2^e 3^f - 1$ *for* $2^e \approx 3^f \approx 2^{2\lambda}$*.*
- $\mathcal{K}$ *is the set of binary strings* $\{0,1\}^e$*.*
- $E_0 \in \mathcal{E}$ *is a starting elliptic curve of unknown endomorphism ring (obtained via trusted setup).*

We remark that the truth of this conjecture is closely related to the truth of Conjecture 1, and the same security considerations apply. The only additional consideration is that we are required to increase our prime size to $p = 2^e 3^f - 1$ in order to have rational $2^e$- and $3^f$-torsion. We require $2^e \approx 2^{2\lambda}$ to guarantee the hardness of computing pre-images and collisions of the unpredictable function, and $3^f \approx 2^{2\lambda}$ to guarantee the hardness of Problem 4, since an adversary who can solve for an isogeny of degree $3^f$ $\psi : E \to E_1$ (or $\psi' : E' \to E'_1$) can distinguish between the two distributions $\mathcal{D}_0$ and $\mathcal{D}_1$. Assumptions of this type go back to the discrete logarithm setting [BNPS03], and in

the isogeny setting, the *auxiliary one-more SIDH* assumption from [BKW20] as well as the *one-more unpredictability* from [Bas24b] in the context of OPRFs, are related to our assumption.[3] In essence, our assumption can be seen as a non-commutative version of the one-more predictability assumptions in [BKW20, Bas24b]. A key and message define coprime kernel generators $K$ and $M$ where the output of the OPRF is $E/\langle M, K\rangle$ and $(E, E/\langle K\rangle, E/\langle M\rangle, E/\langle M, K\rangle)$ describes an SIDH square. Thus it would be sufficient to push a basis on $E$ through to $E/\langle K\rangle$ to break one-more unpredictability, even when this basis is masked. We emphasise that in our setting this is not enough, as it is not in general the case that there is a "parallel" isogeny from $\mathcal{W}(E_0, \mathbf{k})$ to $\mathcal{W}(\mathcal{W}(E_0, \mathbf{m}), \mathbf{k})$, since the secret and message isogenies are not coprime.

In fact, in our case, the evaluator does not even act with the "same" secret $K$ in the sense that it defines a parallel isogeny to its secret isogeny. The algebraic relationship between repeated evaluations of isogenies determined by $\mathbf{k}$ on distinct curves $E_m$ is completely determined by the basis algorithm $\mathcal{B}_N$. Provided $\mathcal{B}_N$ behaves in an unpredictable manner, the evaluator simply applies its secret scalar $k$ to many unrelated bases and reveals the codomain of this action. To further this point, suppose we model $\mathcal{B}_N$ as a random oracle, then the isogeny given by the kernel $\langle \mathbf{k} \cdot \mathcal{B}_N(E_m)\rangle$ admits no algebraic structure, since for all $N$-isogenies $\phi$ originating from $E_m$ there will be a corresponding instantiation of a random oracle that yields a suitable basis such that $\ker \phi = \langle \mathbf{k} \cdot \mathcal{B}_N(E_m)\rangle$. As a consequence, we conjecture that our assumption is at least as hard to break as the hardness assumptions in [BKW20, Bas24b].

We would also like to point out that there exist other, at least semantically related hardness assumptions to ours in the isogeny setting, most notable the *one-more isogeny problem* introduced in the context of the VRF from [Ler25] and of course the *one-more unpredictability* of the CGL hash function-based VRF from Section 5.3. We note that the former reveals 2- or 4-dimensional representations of prime degree isogenies from the same starting curve, while the latter reveals image curves which are directed along paths from $E_m$ to $E_k$ by the same key. Due to the difference in the settings, a direct comparison of the security assumption of [Ler25] and our work is not possible. However, we may view both of our hardness assumptions as special cases of the unpredictable function $\mathcal{W}$ being variants of the CGL hash function.

### 5.4.4 Parameter Selection

In order to make the isogeny evaluations efficient, we need a large rational $2^e$-torsion. For the proof system, we need another large torsion group, which for efficiency reasons, should be a large rational $3^f$-torsion. Thus, we can work over primes of the form

[3] Note that there is an active attack against [BKW20] in [BKM+21] that is mitigated in [Bas24b] by using irrational isogenies. We note that these attacks do not apply to our setting, as there are never any point maps revealed as part of our evaluation.

$p = c2^e3^f - 1$, for $c$ a small co-factor. For the standard attacks not to work, it suffices that $2^e \approx 3^f \approx 2^{2\lambda}$ for a security parameter $\lambda$.

Taking the NIST-level 1 SIKEp434 prime $p = 2^{216}3^{137} - 1$ and setting $\lambda = 128$, we find the following average run times and sizes. The benchmarks were made on a Intel(R) Core(TM) i7-10750H CPU with 2.60GHz. We note that our implementation is not optimised, so we expect some speedup in the running times of our algorithms.

| **KeyGen** | **Eval** | **Verify** | PK size | Proof size |
|---|---|---|---|---|
| 14 ms | 20 sec | 9 sec | 54 bytes | 431 kB |

# Chapter 6

# Cryptanalysis: Flawed Proofs of Knowledge

Cryptographic security proofs are challenging, and even the most well-intentioned cryptographers can miss subtle differences in security properties. This is particularly true in the realm of zero-knowledge proofs, where there are many variants of zero-knowledge and soundness, each possessing subtly different requirements and benefits. Indistinguishability must be defined as either perfect, statistical or computational, which is often not differentiated in poorly written proofs of security. Security properties can be overlooked when they do not apply to the construction in the original paper, but must be taken into account in follow up works, running the risk of being "swept under the rug".

A clear example of this is attempting to construct ring signatures from SQIsign [DKL+20], which requires the careful consideration of the variants of Special Honest Verifier Zero-Knowledge (HVZK). HVZK, loosely speaking, requires that the distributions of a prover's transcripts (who knows the witness for an instance) is indistinguishable from the distribution of a simulator's transcripts (who does not know the witness) when interacting with an honest verifier whose challenge is chosen in advance. An often missed consideration of this property is whether the distinguisher is given access to the witness, which differentiates *strong* and *weak* special HVZK. This differentiation is of particular importance to the security of ring signatures obtained via OR-composition of an underlying proof of knowledge (see [CDS94, Dam10]). More specifically, in the security game for *anonymity against full key exposure*, a distinguisher is given the witness for the special HVZK proof, which may compromise the indistinguishability of transcripts (and thus break anonymity) in the event that the underlying proof is only weak special HVZK. It was shown in [BLL24] that SQIsign is only weak special HVZK, and hence new approaches must be taken to construct ring signatures from this

primitive. In this chapter, we find three examples of flawed cryptographic protocols, the faults of which originate from flawed security proofs for the *zero-knowledge* and *knowledge soundness* properties of a sigma protocol (or in the case of CROSS, a related notion known as a $(q, 2)$-identification scheme).

In the first case, the authors of a variant of the SeaSign signature scheme [Kim24] claim to have eliminated the need for rejection sampling, a technique used to prevent leakage of information about the secret key. In the proof for the zero-knowledge property, they claim that the responses to challenges equal to 0 are independent of the secret key (and thus implied to be simulatable). In Section 6.1, we show that this is not the case, and in fact are biased by the secret key which leads to a key recovery attack. This work was independently published by the author of this thesis in [Lev25a].

In the second case, we return to the classical setting of sigma protocols for proving arithmetic relations on Pedersen commitments. ZKAttest [FLM22] constructs a sigma protocol for users to prove knowledge of a discrete logarithm of a commitment to an elliptic curve point. This has applications in anonymous attestation, for proving knowledge of a valid signature corresponding to an authorised public key [CDH+25]. The protocol suffers from an issue relating to its proof of special soundness. In particular, the extractor does not cover all possible cases of transcripts. We provide a detailed analysis of the soundness issues in Section 6.2 and provide a fix for the protocol in Section 6.3. This work was independently published in [CLR24].

Lastly, in Section 6.4, we consider the identification scheme used to construct the CROSS digital signature [BBB+24]. We point out that they do not satisfy their stated definition of zero-knowledge by constructing a distinguisher to distinguish real and simulated transcripts given access to the witness. Moreover, we show that the real and simulated transcripts are not statistically indistinguishable, and therefore the protocol can only satisfy weak computational (rather than strong, statistical or perfect) Honest Verifier Zero-knowledge. This issue is still present in version 2.0 updated on January 31, 2025, which resolves the security losses attained via the attacks of [BLP+25]. The work in this section was submitted as a note on the IACR ePrint archive [Lev25b].

**Statement of Authorship Contribution (Chapter 6)** *The following section is based on the joint work of [CLR24], and the solo-author works of [Lev25a, Lev25b]. All of the content included in this chapter which was not in my own words has either been rewritten or expanded upon, and I am responsible for the majority of the intellectual contributions of the content present in this chapter. The sections Sections 6.2 and 6.3, from the work of [CLR24], were a collaboration during an internship at Brave research and permission has been granted to include this work in my thesis. Of the co-authors, Celi was responsible for the idea of investigating the work of ZKAttest, and otherwise provided supervisory and writing support. Rowell was responsible solely for the implementation*

*which is not included in this thesis.*

## 6.1 A Key Recovery Attack on a Leaky Variant of Seasign

In this section we discuss a protocol which was vulnerable to a key recovery attack, essentially due to the fact that it is not zero-knowledge. The authors failed to provide a detailed security proof, which may be the cause for this oversight.

SeaSign is an isogeny group-action based signature scheme, first proposed by De Feo and Galbraith [DG19], and later refined by [DPV19]. The scheme is derived from a sigma-protocol for a proof of knowledge of a one-way function obtained by isogeny group-actions. The security of SeaSign relies on rejection sampling, in a Fiat-Shamir-with-aborts [Lyu09] setting, where the prover may restart the protocol in order to prevent leaking information about the secret. The core idea is to ensure that the responses sent by the signer, which are either ephemeral values or differences between ephemeral values and the secret key, remain within specific bounds. If a response falls outside these bounds, the signer aborts the protocol and restarts, thereby preventing any unintended leakage of information about the secret key.

However, the work by Kim [Kim24] introduces a variant of the SeaSign scheme that attempts to bypass the need for rejection sampling, eliminating the potential for unnecessary computations caused by protocol aborts and restarts. The proposed variant claims to achieve this by pre-sampling commitment vectors such that responses will be distributed uniformly for either challenge bit, independent of the secret key. However, we show that this approach inadvertently introduces a bias in the distribution of the responses. The signer's attempt to avoid rejection sampling by pre-sampling commitment vectors leads to a situation where certain responses become impossible, depending on entries of the secret key.

### 6.1.1 The SeaSign Variant in [Kim24]

Since the cryptanalysis in this section does not depend on the technical nature of isogeny-based group actions, we will not delve into the technical details of the SeaSign scheme. Instead, we will briefly describe the relevant features of SeaSign signature generation, and include a description of the variant proposed by Kim [Kim24].

**Group-actions in Isogenies.** When working with elliptic curves and isogenies over a prime field $\mathbb{F}_p$, isogeny computations can be abstracted using the group action framework. For an introduction into group actions from isogenies, we point the interested reader to the original paper by Couveignes [Cou06] as well as to the CSIDH paper [CLM+18] for its instantiation in the supersingular case. In this section, we satisfy ourselves with introducing the group action framework abstractly.

**The SeaSign scheme in [DG19].** SeaSign is based on an identification protocol where the secret key corresponds to a secret vector $\mathbf{e} \in [-B, B]^n$ which is used as input to a one-way function:

$$f(\mathbf{e}) = \mathfrak{i}_1^{\mathbf{e}_1} \cdot \ldots \cdot \mathfrak{i}_n^{\mathbf{e}_n} \star E = E'$$

where $E, E'$ are elliptic curves, and $\mathfrak{i}_i$ are elements of the ideal class group of $\mathrm{End}(E)$, which defines an regular, effective group action on the set of supersingular curves over a finite field $\mathbb{F}_p$. The public key corresponds to $E'$, and the signature is a proof of knowledge that the signer knows the secret key $\mathbf{e}$, with the message tied into the randomness of the challenge computation.

As part of the original protocol, the signer samples, and commits to, random vectors $\mathbf{f}^{(j)} \leftarrow\!\!\$\, [-(\delta+1)B, (\delta+1)B]^n$ for $j \in \{1, \ldots, t\}$. After receiving $t$ single-bit challenges, for each challenge $c_j$, the signer either sends $\mathbf{f}^{(j)}$ if $c_j = 0$ or $\mathbf{f}^{(j)} - \mathbf{e}$ if $c_j = 1$. However, for $c_j = 1$, the signer leaks some information about the vector $\mathbf{e}$, since the distribution of $\mathbf{f}^{(j)} - \mathbf{e}$ is not uniform in $[-(\delta+1)B, (\delta+1)B]^n$. To avoid this, rejection sampling is used. After computing the challenge, if a response vector, which is either $\mathbf{f}^{(j)}$ or $\mathbf{f}^{(j)} - \mathbf{e}$, is not in the bound $[-\delta B, \delta B]^n$, the signer aborts the protocol and restarts. This is repeated until the signer sends a valid response.

**Modifications to Signature Generation in [DPV19].** The approach is refined in the follow up work [DPV19] where aborts are avoided in the case that $c_j = 0$, since this only reveals the ephemeral values $\mathbf{f}^{(j)}$, and leaks nothing about the secret. Furthermore, given $t$ iterations of the sigma protocol, their protocol tolerates up to $u$ aborts (for $u < t$) before the protocol must be re-executed, which substantially improves signature generation efficiency.

**Modifications to Signature Generation in [Kim24].** The approach of [Kim24] differs from these two prior works by attempting to avoid rejection sampling completely, by pre-sampling commitment vectors $\mathbf{f}^{(1)}, \ldots, \mathbf{f}^{(t)}$ such that $\mathbf{f}^{(j)} - \mathbf{e} \in [-\delta B, \delta B]^n$. Since the signer would not need to abort and restart the protocol, this would prevent unnecessary isogeny computations, speeding up signing time. However, the key difference is that the signer cannot know what the challenge bits will be in advance, and cannot prevent bias in the distributions of the responses.

### 6.1.2 Key Recovery Attack

We now state an attack on signatures generated by Algorithm 3 (where $H$ is a secure hash function with $t$-bit output). Suppose that we are given samples

$$\mathbf{f} \leftarrow\!\!\$\, [-(\delta+1)B, (\delta+1)B]^n, \text{ such that } \mathbf{f} - \mathbf{e} \in [-\delta B, \delta B]^n$$

**Algorithm 3** Signature Generation in [Kim24]

**Require:** message $m$, $pk = (E, E_A)$, secret key $\mathbf{e} \in [-B, B]^n$
**Ensure:** $\sigma = (\mathbf{z}^{(1)}, \ldots, \mathbf{z}^{(t)}, c_1, \ldots, c_t)$ ▷ $\mathbf{z}^{(j)} \in [-(\delta+1)B, (\delta+1)B]^n, c_j \in \{0,1\}$

```
cnt ← 1
while cnt ≤ t do
    f^(cnt) ←$ [−(δ+1)B, (δ+1)B]^n
    b ← f^(cnt) − e
    if b ∈ [−δB, δB]^n then                ▷ Resample if out of acceptable bound
        z^(cnt) ← f^(cnt)
        E_cnt ← i_1^{f_1^(cnt)} ... i_n^{f_n^(cnt)} ⋆ E
        cnt ← cnt + 1
    end if
end while
c_1, ..., c_t ← H(j(E_1), ..., j(E_t), m)   ▷ Compute the challenge bits
for j from 1 to t do
    if c_j = 0 then
        z^(j) ← z^(j)                      ▷ The leaky case, if c_j = 0
    else
        z^(j) ← z^(j) − e
    end if
end for
```

for a fixed, uniform secret $\mathbf{e} \in [-B, B]^n$. The first observation is that the $i$-th entries of sampled vectors are uniformly distributed in the set,

$$\mathbf{f}_i \leftarrow\$ [-\delta B + \mathbf{e}_i, \delta B + \mathbf{e}_i],$$

which is clearly a distribution dependent on the secret $\mathbf{e}$. In order to exploit this key dependence, our attack amounts to determining the unknown upper and lower bounds for the distribution above. Suppose you are given $m$ samples $\mathbf{f}^{(1)}, \ldots, \mathbf{f}^{(m)}$. For each $i \in [n]$, we define

$$a_i = \min_{j \in [m]} \left( \mathbf{f}_i^{(j)} + \delta B, B \right) \qquad b_i = \max_{j \in [m]} \left( \mathbf{f}_i^{(j)} - \delta B, -B \right) \tag{6.1}$$

**Theorem 12.** *Given $m$ vectors $\{\mathbf{f}^{(j)}\}_{j \in [m]}$ such that, for all $j \in [m]$ and some fixed $\mathbf{e} \in [-B, B]^n$, it holds that $\mathbf{f}^{(j)} \in [-(\delta+1)B, (\delta+1)B]^n$ and $\mathbf{f}^{(j)} - \mathbf{e} \in [-\delta B, \delta B]^n$. Then for $i \in [n]$ and $a_i$'s and $b_i$'s computed as per Equation (6.1), we have that $\mathbf{e}_i \in [b_i, a_i]$. In particular, if $a_i = b_i$, then $\mathbf{e}_i = a_i = b_i$.*

*Proof.* Suppose that $\mathbf{e}_i \notin [b_i, a_i]$. Since $\mathbf{e}_i \in [-B, B]$, we consider either the case that:

$\mathbf{e}_i < b_i$**:** in which case $b_i \neq -B$ (since $\mathbf{e}_i \geq -B$), and there exists some maximal $\mathbf{f}_i^{(j)}$ such that $\mathbf{f}_i^{(j)} = b_i + \delta B$. Then $\mathbf{f}_i^{(j)} - \mathbf{e}_i = b_i + \delta B - \mathbf{e}_i > \delta B$, which contradicts the assumption that $\mathbf{f}^{(j)} - \mathbf{e} \in [-\delta B, \delta B]$.

$\mathbf{e}_i > a_i$**:** in which case $a_i \neq B$ (since $\mathbf{e}_i \leq B$), and there exists some minimal $\mathbf{f}_i^{(j)}$ such that $\mathbf{f}_i^{(j)} = a_i - \delta B$. Then $\mathbf{f}_i^{(j)} - \mathbf{e}_i = a_i - \delta B - \mathbf{e}_i < -\delta B$, which contradicts the assumption that $\mathbf{f}^{(j)} - \mathbf{e} \in [-\delta B, \delta B]$. □

Hence the attack is as follows. On input of $s$ signatures $\sigma_1, \ldots, \sigma_s$:

1. From each signature, collect the set of vectors $\{z^{(j)}\}$ for which $c_j = 0$. Set $m$ to be the size of this set.
2. For each $i \in [n]$, compute $a_i$ and $b_i$ as per Equation (6.1). Output the set of guesses for $\mathbf{e}$ as $S = \bigoplus_{i=1}^{n} [b_i, a_i]$.
3. For each guess[1] $\mathbf{e} \in S$, check if $\mathbf{e} \star E \stackrel{?}{=} E_A$. If so, output $\mathbf{e}$.

Let $P_{m,i}$ be probability of $m$ biased vectors leaking the $i$-th entry of the secret key, which satisfies the bound:

$$P_{m,i} \geq 1 - 2\left(1 - \frac{1}{2(\delta+1)B+1}\right)^m + \left(1 - \frac{2}{2(\delta+1)B+1}\right)^m$$

Hence the probability $P_m$ of $m$ biased vectors leaking the entire secret key satisfies $P_m = P_{m,i}^n$. The parameters in this setting are not well suited for approximating the binomial terms via low-order Taylor approximations. For reference, however, with parameter set II; $P_{6173} \approx 0.5$. Hence, the attack is expected to recover the secret key in a small number of signatures. We provide practical results of the attack in the next section, which account for the key-space reduction of the signing key.

**On guessing the correct value of secret scalars** By Theorem 12, given $m$ samples for each entry $i \in [n]$, we determine some interval $[b_i, a_i]$ which contains $\mathbf{e}_i$. Fix a row $i$. While it might seem intuitive to prioritise values near the mean value of the interval $[b_i, a_i]$, we show that given samples $\mathbf{f}_i^{(1)}, \ldots, \mathbf{f}_i^{(m)}$, the probability that they result from the distribution $[-\delta B + \mathbf{e}_i, \delta B + \mathbf{e}_i]$ is uniform for the choice of $\mathbf{e}_i$ over $[b_i, a_i]$. By construction of Equation (6.1), we have that $\mathbf{f}_i^{(j)} \in [-\delta B + a_i, \delta B + b_i]$ for all $j \in [m]$. This interval is strictly contained in $[-\delta B + \mathbf{e}_i, \delta B + \mathbf{e}_i]$ if and only if $\mathbf{e}_i \in [b_i, a_i]$. Let $\mathcal{D}_{\mathbf{e}_i}$ be the uniform distribution on $[-\delta B + \mathbf{e}_i, \delta B + \mathbf{e}_i]$. Hence, the event that the

[1] We note that this final step can be performed in $O(\sqrt{S})$ (rather than $O(S)$) group action evaluations by using a meet-in-the-middle/baby-step-giant-step approach.

| | $n$ | $B$ | $\delta$ | $t$ |
|---|---|---|---|---|
| Parameter Set I [DG19] | 74 | 5 | 9472 | 128 |
| Parameter Set II [DPV19] | 74 | 5 | 114 | 337 |

Figure 6.1: Parameter sets used in [Kim24].

observed samples $\mathbf{f}_i^{(1)}, \ldots, \mathbf{f}_i^{(m)}$ arise the distribution $\mathcal{D}_{\mathbf{e}_i}$ is unique when $\mathbf{e}_i \in [b_i, a_i]$ (and cannot occur otherwise). Now, observe that for all $\mathbf{e}_i \in [b_i, a_i]$, it holds that:

$$\begin{aligned}\Pr[\mathbf{e}_i \mid \mathbf{f}_i^{(1)}, \ldots, \mathbf{f}_i^{(m)}] &= \frac{\#\text{Events where } \mathbf{f}_i^{(1)}, \ldots, \mathbf{f}_i^{(m)} \text{ arises from } \mathcal{D}_{\mathbf{e}_i}}{\#\text{Events where } \mathbf{f}_i^{(1)}, \ldots, \mathbf{f}_i^{(m)} \text{arises from } \mathcal{D}_x \text{ for some } x \in [b_i, a_i]} \\ &= \frac{\#\text{Events where } \mathbf{f}_i^{(1)}, \ldots, \mathbf{f}_i^{(m)} \text{ arises from } \mathcal{D}_{\mathbf{e}_i}}{(\#\text{Events where } \mathbf{f}_i^{(1)}, \ldots, \mathbf{f}_i^{(m)} \text{arises from } \mathcal{D}_x) \cdot (\#x \in [b_i, a_i])} \\ &= \frac{1}{1 \cdot (a_i - b_i + 1)}.\end{aligned}$$

This implies there is no better way to guess the correct value of $\mathbf{e}_i$ than to guess uniformly over the interval $[b_i, a_i]$, indicating the attack is in a sense, optimal.

**On avoiding rejection sampling** At a high level, the reason why rejection sampling after the challenge computation cannot be avoided, is that the distribution of responses to a fixed challenge bit must be independent of the secret in both cases. By forcing a bound on either $\mathbf{f}$ or $\mathbf{f} - \mathbf{e}$ prior to knowing which challenge bit is chosen, the signer introduces a bias in the distribution of the responses. In our case, $\mathbf{f} - \mathbf{e}$ was rejected if out of bounds, which induced key dependence to the distributions of responses $\mathbf{f}$ for $c = 0$, but indeed the same issue would occur for responses $\mathbf{f} - \mathbf{e}$ to $c = 1$ if instead the resampling was performed when $\mathbf{f}$ did not satisfy some bound. The signer cannot know the challenge bits in advance, and hence can only perform rejection of leaky responses after challenge computation.

### 6.1.3 Implementation and Benchmarks

We implement the key-recovery attack using a sage script available at https://github.com/levanin/leakysea-public. On each iteration of the experiment, a random key is sampled and a fixed number of biased samples are generated. The protocol of [Kim24] only leaks a biased vector when a challenge bit is 0, which occurs with probability $\frac{1}{2}$. So we will assume that given $s$ signatures with challenge length $t$, we may obtain $\lfloor \frac{st}{2} \rfloor$ biased vectors.

The attack is efficient, and all of our benchmarking was comfortably performed on a laptop over a lunch break. We provide the results of our attack given a varying number of signatures on the parameter sets provided by [Kim24] in Figures 6.2 and 6.3, obtained from the prior works [DPV19, DG19]. Once the key-space has been reduced to a size $2^b$, a meet-in-the-middle search strategy can be used to recover the secret key in time $O(2^{b/2})$, using techniques described in [DG19].

We note that the parameter set I requires a larger number of signatures to effectively perform the attack. This parameter set is designed to handle the high failure probability of the original SeaSign protocol, so ephemeral vectors must be sampled from a much larger space. Hence, there is a lower chance of receiving "good" vectors which leak information about the secret key. We remark that it would have be unreasonable to use these parameters over parameter set II in the first place, since they do not yield any performance benefits over existing work. In particular, the performance of the prior work [DPV19] using parameter set II is roughly $10\times$ faster ($2,195$ s) than the performance of [Kim24] running on parameter set I ($27,685.92$ s), with claimed equivalent security levels.

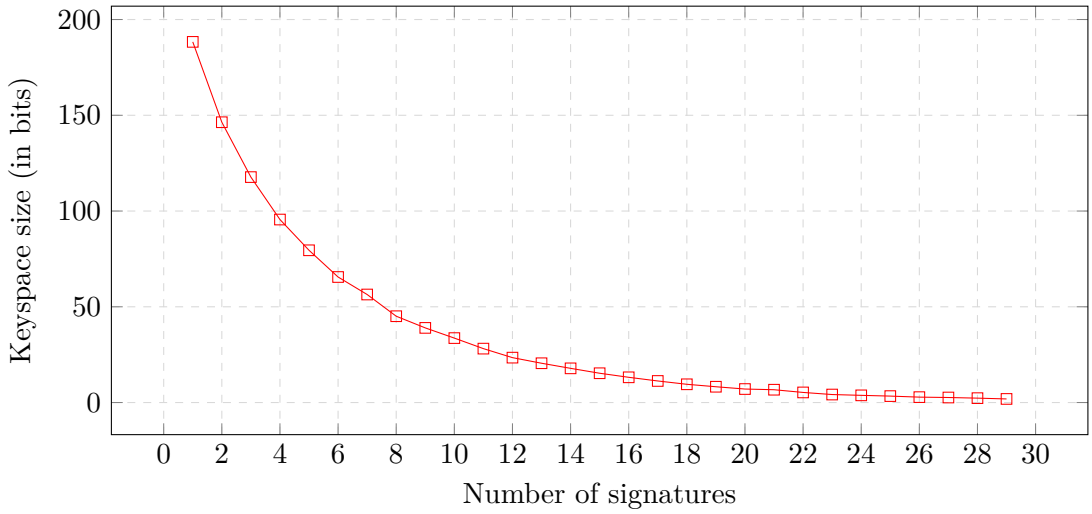



Figure 6.2: Results of our attack on [Kim24, Parameter Set II]. Results are the mean over 100 random instances. The key-space refers to the bit-length of the size of the set of possible secret keys (i.e., if the key-space is $n$ bits, then the number of possible secret keys is $2^n$).

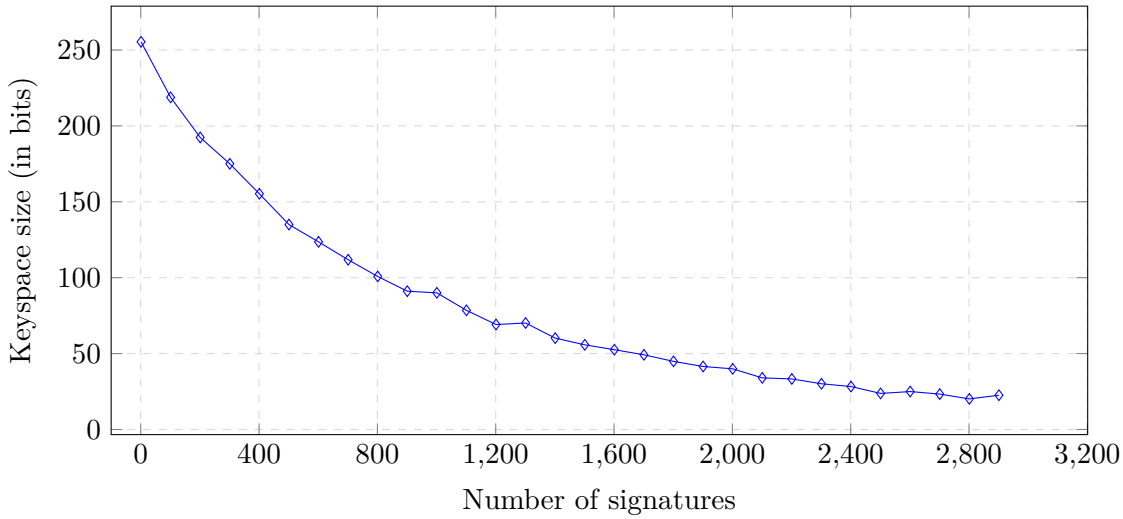



Figure 6.3: Results of our attack on [Kim24, Parameter Set I]. Results are the mean over 20 random instances.

## 6.2 Knowledge Soundness Issues in ZKAttest

ZKAttest is a protocol designed in [FLM22]. The authors introduce the scheme as a way to build both a privacy-preserving ECDSA PoK and a ring signature, which allow a prover who has access to signing functionality (but not necessarily direct access to the private key) to prove knowledge of a valid signature for a given commitment to a public key. Hence, the core primitive in ZKAttest is a non-interactive zero-knowledge proof of knowledge of a valid *ECDSA signature* under a committed public key. Additional properties can be attested, such as proving that the commitment to the public key is a value on a list of valid public keys, resulting in a ring signature.

Under the hood, ZKAttest builds on top of the scheme given by [AGM18], which aims to prove the assertion when working in context of proving arithmetic assertions over elliptic curve groups (since the known techniques for committed discrete logarithm proofs do not work for this case [CS97, NBMV99], as a group element cannot be naturally interpreted as a field element). In the subsections below, we briefly explain the schemes given by [FLM22]. Note that the protocols rely on Figs. 3.1 to 3.3. We point out the steps that were omitted or missing, which we incorporate for clarity and completeness, and then discuss misdesigns and attacks to the constructions.

### 6.2.1 Proof of Affine Point Addition (ZKAttest.PA):

In order to construct a proof of knowledge of a valid ECDSA signature, ZKAttest first introduces a proof that, on input consisting of commitments to three points on an elliptic curve represented in affine coordinates, shows that the sum of the first two points is equal to the third. The formulae for affine point addition (for short Weierstrass curves) are stated in Theorem 13 given by [Sil86].

**Theorem 13** (Point addition)**.** *Let $P := (a_x, a_y), Q := (b_x, b_y) \in E(\mathbb{F}_q)$ be points in affine coordinates, where $E$ is an elliptic curve of short Weierstrass form $E : y^2 = x^3 + ax + b$ for some $a, b \in \mathbb{F}_q$. Given that $P \neq \pm Q$ and $P, Q$ are non-identity elements, the affine point $T := (t_x, t_y) = P + Q$ is given by:*

$$t_x = \left(\frac{b_y - a_y}{b_x - a_x}\right)^2 - a_x - b_x$$

$$t_y = \left(\frac{b_y - a_y}{b_x - a_x}\right)(a_x - t_x) - a_y$$

The above relations given for point addition can be proven by using $\Sigma$-protocol techniques for arithmetic relationships; but, as the point addition formulae is over $\mathbb{F}_q$, the commitments to the coordinates have to be in a group of order $t$. However, it is not generally the case that $\#E(\mathbb{F}_q) = q$. [AGM18] solve this problem by rearranging the point addition formulae so that $\Sigma$-protocols for polynomial relationships among committed values [CM99] and range proofs [Bou00, **?**, BBB+18] can be used on the intermediate commitments. The proof is expanded to handle all the cases for point addition by using OR-composition.

ZKAttest introduces a more practical approach. They resolve the need for range proofs by applying the method given by Bröker and Stevenhagen [BS07] to find elliptic curve groups of prescribed order $q$, in order to instantiate Pedersen commitments over the message space $\mathbb{Z}_q$. The method takes $\tilde{O}(\log q)^3$ steps, and since it need only be run *once* for parameter generation, it is practical, contrary to the comments made in [AGM18]. ZKAttest then prove the relations of Theorem 13 in the protocol below. Note that in the rest of this section and the following, we will write the group operation on Pedersen commitments additively, since they are instantiated via elliptic curves.

**ZKAttest.PA.** Given $C_1 = \mathrm{Com}_q(a_x)$, $C_2 = \mathrm{Com}_q(a_y)$, $C_3 = \mathrm{Com}_q(b_x)$, $C_4 = \mathrm{Com}_q(b_y)$, $C_5 = \mathrm{Com}_q(t_x)$, $C_6 = \mathrm{Com}_q(t_y)$, prove that $T = A + B$, where $A = (a_x, a_y)$, $B = (b_x, b_y)$, $T = (t_x, t_y)$, $(A, B, T) \in E(\mathbb{F}_q)$.

1. The prover computes (note that the verifier can compute $C_7$ and $C_9$ from the public values of $C_1$, $C_2$, $C_3$, $C_4$.):

$$C_7 = C_3 - C_1 = \mathrm{Com}_q(b_x - a_x) \quad C_8 = \mathrm{Com}_q((b_x - a_x)^{-1}),$$
$$C_9 = C_4 - C_2 = \mathrm{Com}_q(b_y - a_y) \quad C_{10} = \mathrm{Com}_q\Big(\frac{b_y - a_y}{b_x - a_x}\Big),$$
$$C_{11} = \mathrm{Com}_q\Big(\Big(\frac{b_y - a_y}{b_x - a_x}\Big)^2\Big) \quad C_{12} = \mathrm{Com}_q(a_x - t_x)\ ,$$
$$C_{13} = \mathrm{Com}_q\Big(\Big(\tfrac{b_y - a_y}{b_x - a_x}\Big)(a_x - t_x)\Big)\ .$$

2. The prover engages with the verifier in the following $\Sigma$-protocols *in parallel* (omitting the prover's input witness values for brevity):

   ⋆ Multiplication proofs via Fig. 3.3:

$$\mathrm{MulProof}(C_7, C_8, \mathrm{Com}_q(1)), \qquad \mathrm{MulProof}(C_8, C_9, C_{10}),$$
$$\mathrm{MulProof}(C_{10}, C_{10}, C_{11}), \qquad \mathrm{MulProof}(C_{10}, C_{12}, C_{13}).$$

   ⋆ Equality proofs via Fig. 3.2:

$$\mathrm{EqualityProof}(C_5, C_{11} - C_1 - C_3), \quad \mathrm{EqualityProof}(C_6, C_{13} - C_2)\ .$$

We highlight some corrections we made to this proof:

- We fix several equations which contain typos.
- We emphasise that all "internal" proofs have to be run in parallel, as is standard in the parallel composition of $\Sigma$-protocols. The implementation of ZKAttest composes them sequentially, which does not yield a sigma protocol and poses security concerns.

Note that this proof assumes that at least two of the points $A, B, T$ are valid points on the curve (see Remark 7), and that $A \neq \pm B$. If the points $A, B$ are randomly chosen, the probability that $A = \pm B$ is $2/|E(\mathbb{F}_q)|$. If this occurs when ZKAttest.PA is invoked in ZKAttest.CDL (see Section 6.2.2), the prover can run the entire protocol again, but technically the protocol is not perfectly complete as is. We stress that in ZKAttest.CDL, this issue can be resolved without the need for extending the point addition proof. Nevertheless, the ZKAttest paper (but not in its implementation) does propose an extension to handle exceptional cases at the cost of efficiency. Note that the protocol guarantees that $((a_x - b_x) \neq 0)$ by verifying it has an inverse in the first inner

multiplication proof. This can be extended to the case when $((a_x - b_x) = 0)$, which corresponds to $P = \pm Q$, and is either a case of point doubling or addition to the point at infinity. theoretically, the authors propose handling of exceptional cases by using AND and OR composition of $\Sigma$ protocols: $(a_x - b_x \neq 0 \wedge t = a + b) \vee (a_x = b_x \wedge a_y = b_y \wedge T = 2P)$ if represented in affine coordinates. This protocol, which would require a subroutine for proving the satisfiability of the doubling formulae (which is unspecified), would be complete provided $A \neq -B$, but still does not account for when $A = -B$, since the points are represented in affine coordinates and the resulting point at infinity $T$ cannot be represented in affine coordinates.

*Remark* 7. Note that in ZKAttest.PA above, and later in Section 6.3.1, it is assumed that at least two of the prover's input points $(A, B, T)$ are indeed valid points on a given curve $E$. If at least two points are valid, then following the point formulae, it follows that the third must be as well. In applications, as the verifier should be convinced of this fact in outer protocols, this seems to be sufficient.

### 6.2.2 Proof of Committed Discrete Logarithm (ZKAttest.CDL):

As stated, [AGM18] builds a proof of the equality of a committed value $\omega$ and the discrete logarithm to the public base point the prover of another committed value $\omega P$. Extending this approach, ZKAttest introduces a ZKAttest.CDL that relies on ZKAttest.PA. For a given instance $(C_1, C_2', C_3')$, the witness is the tuple $(\omega, r_1, r_2, r_3)$ such that $(x, y) = \omega P$, $C_1 = \mathrm{Com}_p(\omega, r_1)$, $C_2' = \mathrm{Com}_q(x, r_2)$ and $C_3' = \mathrm{Com}_q(y, r_3)$. The protocol can be seen below, where solid boxes represent the values sent by a party.

**ZKAttest.CDL.** Given $C_1 = \mathrm{Com}_p(\omega) = \omega P + r_1 Q$, $C_2' = \mathrm{Com}_q(x) = xP' + r_2 Q'$, $C_3' = \mathrm{Com}_q(y) = yP' + r_3 Q'$, for $q$ equal to the modulus of the base field of $E$, prove that $S = (x, y)$ is equal to $\omega P$, where $P, Q \in E$ are public elements of prime order $p$, and $(P', Q')$ are points in $E'$ of prime order $q$.

1. The **prover**::

   - ⋆ chooses a random $\alpha, \beta_1 \in \mathbb{Z}_p$, and $\beta_2, \beta_3, \ \beta_4, \beta_5 \ \in \mathbb{Z}_q$,
   - ⋆ sets $(\gamma_1, \gamma_2) = \alpha P$, and
   - ⋆ sets $(u, v) = (\alpha - \omega)P$

   They, then, compute the following values:

$$a_1 = Com_p(\alpha) = \alpha P + \beta_1 Q,$$
$$a_2 = Com_q(\gamma_1) = \gamma_1 P' + \beta_2 Q', \qquad a_3 = Com_q(\gamma_2) = \gamma_2 P' + \beta_3 Q',$$
$$C_4' = Com_q(u) = uP' + \beta_4 Q', \qquad C_5' = Com_q(v) = vP' + \beta_5 Q'.$$

sending $\boxed{a_1, a_2, a_3, C'_4, C'_5\ , \texttt{comm}'}$ to the verifier as `comm`,
where `comm`′ is for ZKAttest.PA on $(C'_2, C'_3, C'_4, C'_5, a_2, a_3)$.

2. The **verifier**:

   - chooses a challenge string $c = (c_0, c_1)$, where $c_0$ is a single random bit $\in \{0, 1\}$ and $c_1 \in \mathbb{Z}_q$ is a challenge for the ZKAttest.PA.

   They send $\boxed{c}$ as `chall`.

3. The **prover** receives $c$:

   ⋆ If $c_0 = 0$, computes $z_1 = \alpha$, $z_2 = \beta_1$, $z_3 = \beta_2$, $z_4 = \beta_3$.
   Sends the tuple $\boxed{(z_1, z_2, z_3, z_4)}$ as `resp`.

   ⋆ If $c_0 = 1$, computes $z_1 = \alpha - \omega$, $z_2 = \beta_1 - r_1$, $z_3 = \beta_4$ , $z_4 = \beta_5$ . Then, they compute the response for ZKAttest.PA:
   - Given $T = z_1 P = (u, v)$,
   - Compute `resp`′ with `chall`′ $= c_1$. which verifies that $T = (\gamma_1, \gamma_2) - (x, y)$ $(T = \alpha P - S)$.

   Sends the tuple $\boxed{(z_1, z_2, z_3, z_4\ , \texttt{resp}'\ )}$ as `resp`.

4. Upon receiving `resp`, the **verifier** performs the following:

   ⋆ If $c_0 = 0$, computes $(t_1, t_2) = z_1 P$. Then, verifies that $a_1 \stackrel{?}{=} z_1 P + z_2 Q$, $a_2 \stackrel{?}{=} Com_q(t_1, z_3)$ and $a_3 \stackrel{?}{=} Com_q(t_2, z_4)$.

   ⋆ If $c_0 = 1$, computes $(t_1, t_2) = z_1 P$. Then, verifies that $a_1 \stackrel{?}{=} z_1 P + z_2 Q + C_1$, that $C'_4 \stackrel{?}{=} Com_q(t_1, z_3)$ and $C'_5 \stackrel{?}{=} Com_q(t_2, z_4)$ , and sequentially verifies the point addition proof $\pi = (\texttt{comm}', c_1, \texttt{resp}')$.

It is worth noting that, as pointed out by ZKAttest, the proof from [AGM18] fails to verify $C_1$, the commitment to the secret value $\omega$. ZKAttest presents the corrected version of this proof with this verification. But even with this correction, the proof does not achieve special soundness. In short, this is due to the non-standard approach used in the AND-composition of sigma protocols.

Below, we discuss the issues with ZKAttest's proof of security for this PoK. We note that the PoK is indicated to be a $\Sigma$-protocol and it is specified (by the number of repetitions) to have a knowledge error of $\frac{1}{2}$, and perfect HVZK. We explore the failures of the properties that ZKAttest.CDL should provide.

**Completeness.** We remark that the protocol ZKAttest.CDL does not satisfy perfect completeness. If $\alpha \in \{0, \omega, 2\omega\}$, then the inner ZKAttest.PA will fail, since it does not handle point doubling, inverse addition (without its extension), and addition by the identity. This failure, in turn, means that the outer protocol will fail, and we note that these exceptional cases occur with probability $\frac{3}{p}$. A solution to this completeness issue would be to have an honest prover sample from $\mathbb{Z}_p \backslash \{0, \omega, 2\omega\}$. As a trade-off, doing so leads to a protocol that is *statistical* honest verifier zero-knowledge (with $\text{negl}(p)$ statistical closeness between real and simulated transcripts).

**Special Honest Verifier Zero-Knowledge.** The proof of SHVZK provided in the final version of ZKAttest is missing details, but it can be shown that their protocol satisfies statistical SHVZK, as they state, using a simulator similar to the one given in the proof of Theorem 15.

**Special Soundness.** The proof of special soundness in ZKAttest is flawed due to inherent misdesign. First, note that in the original protocol specification and implementation the prover does not construct the commitment phase for the point addition protocol until after the challenge has been received[2], and their implementation computes challenges for the subroutines of the point addition proof on the fly. Sending a challenge before the commitment phase leads to a trivial attack where one can forge a valid point addition proof in the same way the HVZK simulator for ZKAttest.PA behaves. In their published version, they correctly include commitment phase for ZKAttest.PA in the first round, but this issue is present in their open source implementation at the time of publication.

The authors of ZKAttest claim the protocol is 3-special-sound, and construct an extractor which takes as input three transcripts, $(\texttt{comm}_i, \texttt{chall}_i, \texttt{resp}_i)_{i\in[3]}$, where $\texttt{chall}_1 = (0, a)$, $\texttt{chall}_2 = (1, b)$, $\texttt{chall}_3 = (1, c)$ with $b \neq c$ and $a, b, c \in \mathbb{Z}_q$. Such an accepting transcript would allow for the extraction of the witness since:

- Given that ZKAttest.PA is 2-special sound, the extractor can invoke the extractor of this internal proof with transcripts for the challenges $b, c$. This would yield $x, y, r_2, r_3$ as output.
- the extractor may take $\texttt{resp}_1 = (z_1, z_2, z_3, z_4)$ and $\texttt{resp}_2 = (z_1', z_2', z_3', z_4', \pi)$, and deduce $\omega = z_1 - z_1'$ and $r_1 = z_2 - z_2'$.

We note that for this specific input, the witness extracted above is valid. For further justification, see the proof of the fixed protocol in Section 6.3.2. However, the extractor

[2]This is likely also the case in [AGM18], however the specification of the protocol is not sufficiently detailed.

is still flawed. In particular, there is no way to extract the witness given the following cases with these transcript triples (with $a, b, c \in \mathbb{Z}_q$):

1. $(\texttt{comm}_i, \texttt{chall}_i, \texttt{resp}_i)_{i\in[3]}$ where $\texttt{chall}_1 = (0, a)$, $\texttt{chall}_2 = (0, b)$, $\texttt{chall}_3 = (1, c)$ for $a \neq b$. We explain this extractor fault first since it is the easiest to correct. In this case, the extractor may recover the values $\omega, x, y, r_1$, but not the randomness $r_2, r_3$. In particular, if we modify the protocol to run ZKAttest.PA independently of $c_0$, such that the prover engages in it for both $c_0 \in \{0, 1\}$, then this case can yield a valid witness extraction.

2. $(\texttt{comm}_i, \texttt{chall}_i, \texttt{resp}_i)_{i\in[3]}$ where $\texttt{chall}_1 = (1, a)$, $\texttt{chall}_2 = (1, b)$, $\texttt{chall}_3 = (1, c)$ for $a \neq b \neq c$. In this case, the extractor may recover part of the witness by invoking the extractor of ZKAttest.PA, but the extractor cannot recover $\omega$, only the openings of $C'_2, C'_3$. There is no clear solution that allows the extractor to recover $\omega$ in this setting.

3. $(\texttt{comm}_i, \texttt{chall}_i, \texttt{resp}_i)_{i\in[3]}$ where $\texttt{chall}_1 = (0, a)$, $\texttt{chall}_2 = (0, b)$, $\texttt{chall}_3 = (0, c)$ for $a \neq b \neq c$. In this case, it follows that the extractor can learn nothing about the witness, since $\texttt{resp}_1 = \texttt{resp}_2 = \texttt{resp}_3$. If we perform the same modification to the proof as in the first case, we still remain with the same issue as in the second case, where it is not possible to extract $\omega$.

Note that the definition of 3-special soundness requires the extractor to succeed in extracting the witness for *any* 3 accepting transcripts. Therefore, the scheme is not 3-special sound. Furthermore, the claim that the protocol above has soundness error $\frac{1}{2}$ is left unjustified. Recall that an $n$-special sound protocol of challenge space $C$ has knowledge error $\frac{n-1}{|C|}$. If the scheme was 3-special sound, it would have knowledge error $\frac{2}{2q}$ (since $C = \mathbb{Z}_2 \times \mathbb{Z}_q$), which is unrealistic for this construction.

### 6.2.3 A Practical Attack on ZKAttest's Implementation

In addition to the above concerns, the authors of ZKAttest implemented the non-interactive protocol with the Fiat-Shamir transform applied on 128 repetitions. However, as an efficiency measure (as stated in Section 8 of their paper), the verification is only performed on a random subset of 20 of the repetitions. This ad-hoc choice reduces the probability of a forged proof being accepted to at least $2^{-20}$ (we note this bound is not tight due to the soundness issues above), but may also allow for a further reduction to security. Note that:

A malicious prover may construct the commitment phase for $c_0 = 0$ in the same fashion as the HVZK simulator, $\mathcal{S}$, for the 128 repetitions, and compute the resultant challenge, which is a hash of the concatenation of these commitments. Then, they

arbitrarily select the transcript of a single repetition. They will replace the commitment to $C_4'$ in this repetition in order to change the output of the resulting challenge hash. This can be done as this commitment is never opened for the verifier and should be uniformly distributed in the point-set of $E'(\mathbb{F}_{q'})$. For $i$ steps, the malicious prover sets $C_4'$ to a random point[3] in $E'(\mathbb{F}_{q'})$ until the resultant hash yields a challenge string which contains at least $m$ challenges which have $c_0 = 0$. Call these the 'good' challenges. For the 'good' challenges, they complete their proofs as in the simulator, and for 'bad' challenges they output uniformly random values as response.

For $m = 115$, the probability that a verifier chooses repetitions which have 'good' challenges, and accepts the forged proof, is roughly 2%. We make some heuristic assumptions as to the practicality of this attack. The expected number of attempts to find a hash which has at least 115 zeroes is $i \approx 2^{70}$. This takes $2^{70}$ hash computations. Now, assuming each increment's hash only requires a single SHA-256 execution and basing the cost of SHA-256 computations on the revenue of Bitcoin mining[4], the approximate cost of the attack is 1500 USD.

Even with the counter-measure of rate-limiting, this "cheap" attack is sufficiently practical if launched in a distributed fashion. Alongside the other issues, the authors of ZKAttest have been made aware and acknowledged this attack.

## 6.3 Sound Proofs of Knowledge: CDLS

In the following sections, we consider several solutions to the issues faced with the flawed security proof of ZKAttest's ZKAttest.CDL for proving knowledge of an elliptic curve discrete logarithm. Recall that in our context, we are interested in protocols which operate in the elliptic curve setting, where [NBMV99] does not apply.

In Section 6.3.1, we propose an optimised Σ-protocol for proving that a commitment to an elliptic curve point is the sum of two others, which we call CDLS.PA. In Section 6.3.2, we propose a Σ-protocol with knowledge error $\frac{1}{2}$, for a proof of committed discrete logarithm, which we call CDLS.CDL, or just CDLS if the context is implicit. This can be used instead of ZKAttest to construct a secure NIZKPoK of a valid ECDSA signature verification under a committed public key. Then, in the same fashion as ZKAttest.CDL, the NIZKPoK can be composed with a Σ-protocol for proving set membership of a committed key in order to construct a ring signature using the generic construction of [GK15].

[3] Using the common compressed representation for elliptic curve points, which is the $x$-coordinate along with a parity bit, one does not need to evaluate the curve equation each time.

[4] See https://charts.woobull.com/bitcoin-hash-price/, which as of 2023 places the value of $10^{12}$ SHA-256 hashes at approximately $10^{-6}$ USD, assuming modern ASICs can be set up to handle arbitrary fixed length input.

### 6.3.1 Proof of Valid Point Addition (CDLS.PA)

A key insight that yields optimisation beyond [CM99, FLM22, AGM18], as discussed in Section 3.5, is that only multiplication and a small number of opening proofs are necessary in the process of proving a polynomial relation. In particular, linear combinations of commitments can be obtained without any interaction from the prover. As an example, consider a prover who wishes to convince the verifier that taking the product of the opening of two commitments $X = \texttt{comm}(x)$, $Y = \texttt{comm}(y)$ is a linear combination of $n$ other committed values, such that for $Z_i = \texttt{comm}(z_i)$, $i \in [n]$,

$$xy = \sum_{i \in [n]} a_i z_i$$

In this example, the prover may verify the relationship holds by having verifier compute $T = \prod_{i \in [n]} Z_i^{a_i}$, and engaging them with $\text{MulProof}(X, Y, T)$. This holds since $T$ is a valid commitment for the linear combination. The opening proofs are needed to allow an extractor to recover the openings for the individual commitments given an opening for the linear combination.

**CDLS.PA.** Given $C_1 = \text{Com}_q(a_x)$, $C_2 = \text{Com}_q(a_y)$, $C_3 = \text{Com}_q(b_x)$, $C_4 = \text{Com}_q(b_y)$, $C_5 = \text{Com}_q(t_x)$, $C_6 = \text{Com}_q(t_y)$, prove that $T = A + B$, where $A = (a_x, a_y)$, $B = (b_x, b_y)$, $T = (t_x, t_y)$, $(A, B, T) \in E(\mathbb{F}_q)$.

Recall the elliptic curve addition formulae (stated in Theorem 13). We may rearrange this formula into a system of equations, by adding an additional variable $\tau$ (note that $\tau = \frac{b_y - a_y}{b_x - a_x}$):

$$(b_x - a_x)\tau = b_y - a_y \tag{6.2}$$

$$\tau^2 = a_x + b_x + t_x \tag{6.3}$$

$$\tau(a_x - t_x) = a_y + t_y \tag{6.4}$$

With this rearrangement, the prover will send the commitment $C_7 = \text{Com}_q(\tau)$ along with the previously defined commitments $(C_1, \ldots, C_6)$ in the commitment phase of CDLS.PA. The prover engages with the verifier on the instance $(C_1, \ldots, C_6)$. Note that the verifier can compute the commitment to any linear combination of known commitments (including $C_7$) due to the linearity of Pedersen commitments. They perform the following proof interactions in AND-composition (with a common single challenge in $\mathbb{Z}_q$):

- $\text{MulProof}(C_3 - C_1, C_7, C_4 - C_2)$ which verifies that Eq. (6.2) holds,
- $\text{MulProof}(C_7, C_7, C_1 + C_3 + C_5)$ which verifies that Eq. (6.3) holds,

- MulProof$(C_7, C_1 - C_5, C_2 + C_6)$, which verifies that Eq. (6.4) holds.
- OpeningProof$(C_2)$, which allows the extractor to fully recover a witness.
- NonZeroProof$(C_3 - C_1)$, which verifies that the $b_x - a_x \neq 0$ (i.e. $A \neq \pm B$).

We note that without the final non-zero check, the prover may maliciously choose points $A, B$ such that $A = -B$. This unconstrains the value of $\tau$, and thus the prover can choose any $t_x, t_y$ and $\tau$ such that Eqs. (6.3) and (6.4) hold.

As an abuse of notation, we write the elliptic curve group operations above additively, instead of multiplicatively as in Figs. 3.1 and 3.3. The verifier accepts if all of the above protocols are accepting.

**Theorem 14.** *The protocol described above is a $\Sigma$-protocol for the relation* $\mathcal{R} =$

$$\left\{ \left( \begin{pmatrix} C_1, C_2, \\ C_3, C_4, \\ C_5, C_6 \end{pmatrix}, \begin{pmatrix} a_x, a_y, b_x, b_y, \\ t_x, t_y, r_1, r_2, \\ r_3, r_4, r_5, r_6 \end{pmatrix} \right) \;\middle|\; \begin{array}{c} A + B = T \text{ where } A \neq \pm B, A, B \neq \mathcal{O} \\ A = (a_x, a_y), B = (b_x, b_y),\ T = (t_x, t_y) \\ \text{Each commitment } C_i \text{ is valid} \\ \text{with randomness } r_i \end{array} \right\}$$

*assuming the coordinates of at least two of the points correspond to valid points on an elliptic curve.*

*Proof.* We show the protocol satisfies *completeness*, *honest verifier zero-knowledge* and *special-soundness*.

**Completeness.** Due to the correctness of the point addition formulae, the multiplication proofs will be accepted, since the coordinates of the points must satisfy Eqs. (6.2) to (6.4), given that $A \neq \pm B$, and neither points correspond to the identity element. It is worth highlighting the following two cases: i) if $A$ or $B$ are maliciously chosen such that $A = \pm B$, the verifier will reject the non-zero check. ii) if $A, T$ are chosen such that $A = \pm T$, then Eq. (6.4) implies that $a_y = -t_y$ and hence that $A = -T$, which is a valid case (in particular, $B = -2A$). Since the system of equations is symmetric in the choice of $A$ and $B$, the same holds for $B = \pm T$.

**Honest Verifier Zero-Knowledge.** On input $(C_1, \ldots, C_6)$, and challenge $c$, the simulator samples random $(a, b) \leftarrow\!\!\$\ [q]$, and sets $C_7' = \mathrm{Com}_q(a, b)$, adding it to the commitment phase of the transcript. The simulator then invokes the simulators for the three inner multiplication proofs and opening proof where $C_7'$ is used as input in lieu of $C_7$. The scheme is perfect HVZK due to the perfect hiding property of the commitment $C_7'$, and the perfect HVZK of the underlying inner multiplication and opening proofs.

**2-Special Soundness.** The proof is constructed by AND-composing the relations for the system of equations realising point addition, and hence soundness should hold by construction. Nevertheless, we will explicitly show how to extract a valid witness given the extractors for the sub-protocols MulProof, OpeningProof and NonZeroProof.

Recall that, given colliding transcripts for the protocol $(\texttt{comm}, \texttt{chall}, \texttt{resp})$ and $(\texttt{comm}, \texttt{chall}', \texttt{resp}')$, we have colliding transcripts for each of the 5 AND-composed protocols (with common challenges $\texttt{chall}, \texttt{chall}'$). The extractor invokes the sub-extractor for the three multiplication proofs, respectively. In particular, the extractor learns the quantities $b_x - a_x$, $a_x + b_x + t_x$ and $a_x - t_x$. Note that the extractor also recovers $b_y - a_y$, $a_y + t_y$, $r_4 - r_2$ and $r_2 + r_6$ from the extractors of the multiplication proofs.

To recover the $x$-coordinates and associated randomness, the extractor solves a system of 3 linear equations in three unknowns and recovers $a_x$, $b_x$, and $t_x$. The randomness $r_1, r_3, r_5$, which satisfies the same system of equations, is extracted similarly.

To recover the $y$-coordinates and associated randomness, the extractor invokes the sub-extractor for the opening proof of $C_2$, learning $a_y$ and $r_2$. Having the $y$ coordinate and the associated randomness, the extractor can recover $b_y, t_y$ and $r_4, r_6$ by substituting the known values for $a_y$ and $r_2$.[5]

Note that the committed values satisfy the affine point coordinate equations, since $A \neq \pm B$ by the non-zero check. Further, neither $A$ nor $B$ can correspond to the identity since they are represented in affine coordinates. Hence, the extractor recovers a valid witness.

□

### 6.3.2 A Fixed Proof of Committed Discrete Logarithm (CDLS.CDL)

Due to the flawed extractor in ZKAttest.CDL, we propose reducing the challenge space of the inner point addition protocol. The intuition behind this choice, is that the outer ZKAttest.CDL has knowledge error of at least $\frac{1}{2}$, and thus the inner ZKAttest.PA (which has knowledge error of $\approx \frac{1}{q}$) cannot be utilised properly. Moreover, instead of running the point addition proof conditionally on the response of the verifier, we correctly compose the protocols via AND-composition. Lastly, we also differentiate between the base of the discrete logarithm, $R$, and the parameter the prover used as a parameter in the pedersen commitment scheme, as these need not necessarily be equal.

[5] Note that while it is possible to recover the $y$-coordinates given only the extracted witnesses for the multiplication proofs: it requires evaluating the $x$-coordinates through the curve equation and deducing the correct choices of sign by the known quantities. This does not allow for the recovery of associated randomness, which is why we include the opening proof for $C_2$.

**CDLS.CDL**. Given $C_1 = \mathrm{Com}_p(\omega) = \omega P + r_1 Q$, $C'_2 = \mathrm{Com}_q(x) = xP' + r_2 Q'$, $C'_3 = \mathrm{Com}_q(y) = yP' + r_3 Q'$, for $q$ equal to the modulus of the base field of $E$, prove that $S = (x, y)$ is equal to $\omega R$, where $R, P, Q \in E(\mathbb{F}_q)$ are public points of prime order $p$; $P', Q' \in E'(\mathbb{F}_{q'})$ are public points of prime order $q$ and $(P, Q)$, $(P', Q')$ instantiate $\mathrm{Com}_p$ and $\mathrm{Com}_q$ respectively.

1. **The prover**:
   (a) chooses a random $\alpha, \beta_1 \in \mathbb{Z}_p$, and $\beta_2, \beta_3, \beta_4, \beta_5 \in \mathbb{Z}_q$, such that $\alpha \notin \{0, \omega, 2\omega\}$.
   (b) sets $(s, t) = \alpha R$ (the $x$ and $y$ coordinates of $\alpha R$),
   (c) sets $(u, v) = (\alpha - \omega)R$ (the $x$ and $y$ coordinates of $((\alpha - \omega)R)$),

   Then, they compute commitments to $\alpha$, and to the coordinates of $\alpha R$ and $((\alpha - \omega)R)$ $((s, t), (u, v)$, respectively) in the following way:

$$C_4 = \mathrm{Com}_p(\alpha) = \alpha P + \beta_1 Q,$$
$$C'_5 = \mathrm{Com}_q(s) = sP' + \beta_2 Q', \qquad C'_6 = \mathrm{Com}_q(t) = tP' + \beta_3 Q',$$
$$C'_7 = \mathrm{Com}_q(u) = uP' + \beta_4 Q', \qquad C'_8 = \mathrm{Com}_q(v) = vP' + \beta_5 Q'.$$

   Then, they send $C_4, C'_5, C'_6, C'_7, C'_8$ to the verifier. In parallel, they send the commitments for the inner CDLS.PA (as in Section 6.3.1), with binary challenge space, and input $(C'_2, C'_3, C'_7, C'_8, C'_5, C'_6)$ that will be used to verify that $\alpha R = \omega R + ((\alpha - \omega)R)$.

2. **The verifier** chooses a random challenge $c \in \{0, 1\}$ and sends it to the prover.

3. **The prover** receives $c$ and performs the following:
   (a) If $c = 0$, sends $(z_1, z_2, z_3, z_4) = (\alpha, \beta_1, \beta_2, \beta_3)$.
   (b) If $c = 1$, sends $(z_1, z_2, z_3, z_4) = (\alpha - \omega, \beta_1 - r_1, \beta_4, \beta_5)$.
   (c) They send the response to the inner CDLS.PA with challenge $c$.

4. Upon response, **the verifier** performs the following:
   (a) If $c = 0$, computes $(s', t') = z_1 R$ and check that $C_4 \stackrel{?}{=} \mathrm{Com}_p(z_1, z_2)$, $C'_5 \stackrel{?}{=} \mathrm{Com}_q(s', z_3)$ and $C'_6 \stackrel{?}{=} \mathrm{Com}_q(t', z_4)$.
   (b) If $c = 1$, computes $(u', v') = z_1 R$ and check that $C_4 - C_1 \stackrel{?}{=} \mathrm{Com}_p(z_1, z_2)$, $C'_7 \stackrel{?}{=} \mathrm{Com}_q(u', z_3)$ and $C'_8 \stackrel{?}{=} \mathrm{Com}_q(v', z_4)$.
   (c) In parallel, verifies the response for the point addition proof with binary challenge $c$.

**Theorem 15.** *CDLS.CDL is a $\Sigma$-protocol for the relation* $\mathcal{R} =$

$$\left\{ ((C_1, C_2', C_3'), (\omega, r_1, r_2, r_3)) \;\middle|\; \begin{array}{c} (x,y) = \omega R,\ C_1 = Com_p(\omega, r_1), \\ C_2' = Com_q(x, r_2),\ C_3' = Com_q(y, r_3) \end{array} \right\}$$

*assuming $\omega \neq 0$ and the instantiations of the Pedersen commitment schemes $Com_p$, $Com_q$ are perfectly hiding and computationally binding.*

*Proof.* We aim to prove *completeness*, *HVZK* and *special soundness*.

**Completeness.** If the prover knows the witness $w$, and samples an $\alpha$ such that $\alpha - \omega \notin \{0, \omega, -\omega\}$, then the inner CDLS.PA will accept with probability 1, the equalities in Step 4 will hold, and $\mathcal{V}$ will always accept.

**Honest Verifier Zero-Knowledge.** We construct a simulator $\mathcal{S}$ for an accepting transcript which is *statistically indistinguishable* from a real accepting transcript. On input challenge $c$, the simulator does the following:

- If $c = 0$, the simulator randomly samples $z_1 \in \mathbb{Z}_p \backslash \{0\}$, $z_2 \in \mathbb{Z}_p$, $z_3, z_4, u, v \in \mathbb{Z}_q$, and sets $(s,t) = z_1 R$. The simulator computes $C_4 = z_1 P + z_2 Q$, $C_5' = sP' + z_3 Q'$ and $C_6' = tP' + z_4 Q'$, and sets $C_7', C_8'$ as commitments to the random values $(u, v)$.

- If $c = 1$, the simulator randomly samples $z_1 \in \mathbb{Z}_p \backslash \{0\}$, $z_2 \in \mathbb{Z}_p$ and $z_3, z_4, s, t \in \mathbb{Z}_q$, and sets $(u,v) = z_1 R$. The simulator computes $C_4 = z_1 P + z_2 Q + C_1$, $C_7' = uP' + z_3 Q'$, $C_8' = vP' + z_4 Q'$ and $C_5', C_6'$ as the commitments to the random values $(s,t)$.

In both cases, the simulator invokes the simulator for CDLS.PA on input $(C_2', C_3', C_7', C_8', C_5', C_6')$ and the binary challenge $c$. We show that real and simulated transcripts are *statistically indistinguishable* given that the simulator of CDLS.PA outputs a transcript that is perfectly indistinguishable from a real transcript. Observe that if $c = 0$ (resp. $c = 1$), $z_1$ in the real transcript is a uniformly random value in $\mathbb{Z}_p \backslash \{0, \omega, 2\omega\}$ (resp. $\mathbb{Z}_p \backslash \{0, -\omega, \omega\}$) and $z_1$ in the simulated transcript is a uniformly random value in $\mathbb{Z}_p \backslash \{0\}$. Call the real sampling distribution $X_q$ and the simulated sampling distribution $Y_q$ (resp. $Y_q'$). Note

that the statistical distance:

$$\begin{aligned}\Delta(X_q, Y_q) &= \frac{1}{2}\sum_{x\in\mathbb{Z}_q\setminus\{0\}} |\Pr[X_q = x] - \Pr[Y_q = x]| \\ &= \frac{1}{2}\Big(2|0 - \frac{1}{q-1}| + \sum_{x\in\mathbb{Z}_q\setminus\{0,\omega,2\omega\}} |\frac{1}{q-3} - \frac{1}{q-1}|\Big) \\ &= \frac{1}{q-1} + \frac{1}{q-1} \\ &= \frac{2}{q-1} \quad (= \Delta(X_q, Y'_q) \text{ by a similar argument})\end{aligned}$$

is statistically indistinguishable, since we require that $q = \exp(\omega)$, which is negligible in $\omega$. Furthermore, both in the real and simulated proofs, $z_2, z_3, z_4$ is uniformly random in their respective domains ($z_2 \in \mathbb{Z}_p$ and $z_3, z_4 \in \mathbb{Z}_q$). If $c = 0$ (resp $c = 1$), the verification equations uniquely determine $C_4, C'_5, C'_6$ (resp. $C_4, C'_7, C'_8$) conditioned on $(z_1, z_2, z_3, z_4, C_1)$ and that the remaining commitments are to uniformly random inputs in $\mathbb{Z}_q$. Since the commitment scheme is perfectly hiding, the commitments in the real and simulated transcripts are perfectly indistinguishable. Hence, the real and simulated transcripts are statistically indistinguishable with statistical distance $\frac{2}{q-1}$.

**2-Special Soundness.** Given two accepting transcripts for the protocol for challenges $c = 0$ and $c' = 1$, the extractor invokes the extractor for the sub-protocol CDLS.PA, which renders $r_2, r_3$ as openings to the commitments $C'_2, C'_3$.

Let $(z_1, z_2, z_3, z_4), (z'_1, z'_2, z'_3, z'_4)$ be the responses for $c$ and $c'$, respectively. By the verification equations, we know that $C'_5, C'_6$ and $C'_7, C'_8$ are valid commitments to the coordinates of points $z_1P$ and $z'_1P$, respectively. Furthermore, since both the transcripts are accepting, CDLS.PA must correspond to a valid instance, and thus we know that the commitments to the coordinates of $\omega R$, $z_1R$ and $z'_1R$ must satisfy the equation $\omega R + z'_1R = z_1R$. Lastly, we know by the verification equations, that $C_4 - C_1 = \mathrm{Com}_p(z'_1, z'_2)$ and $C_4 = \mathrm{Com}_p(z_1, z_2)$, and hence we can recover the opening to $C_1$ as $\omega = z_1 - z'_1$, and $r_1 = z_2 - z'_2$. Hence, the extractor recovers $\omega, r_1, r_2, r_3$. By the point addition proof, and the consistency of the commitments of $z_1 = \alpha, z'_1 = \alpha - \omega$, we must have that for $(x, y) = (z_1 - z'_1)P$, $C'_2 = \mathrm{Com}_q(x, r_2)$ and $C'_3 = \mathrm{Com}_q(y, r_3)$. □

*Remark* 8. Constructing a 5-round protocol was considered, where in the first round, the prover would open one of the points and verify the consistency of the point multiplication commitments. In the second, the prover would run the point addition protocol (conditional on the first challenge being 1). We believe such an interactive protocol would be secure, with the techniques described in [AFK22]. However, in the analysis of [AFK22], the authors claim knowledge error loss when the Fiat-Shamir transform is

applied to parallel repetitions of a multi-round protocol. In this case, the security loss would be quadratic in the number of random oracle queries of an attacker. Since the 5 round protocol would offer very few benefits if provably secure as a $(2, q)$-special-sound protocol, with expected proof lengths roughly $\frac{1}{3}$ shorter than the protocol above in the best case, we opt to remain in the more flexible and secure setting of 3-round protocols.

#### 6.3.2.1 Transforming to Non-Interactive Zero-Knowledge Proof of Knowledge

To boost the soundness of CDLS.CDL, we run $\lambda$ parallel repetitions, and apply[6] the Fiat-Shamir transform [FS87] to obtain a NIZKPoK with knowledge error of $2^{-\lambda}$. There is still a healthy margin for the statistical zero-knowledge parameter. Given $\lambda$ repetitions, a real and simulated transcript can be distinguished by an unbounded distinguisher with advantage at most $\frac{2\lambda}{2^{2\lambda}-1} \leq 2^{-\lambda}$ (for $\lambda \geq 3$).

Let the commitments for $\mathbb{Z}_q$ (resp. $\mathbb{Z}_p$) correspond to points on an elliptic curve $E'$ (resp $E$) defined over a field of $q'$ (resp. $q$) elements. Assume[7] $\log_2 p \approx \log_2 q \approx \log_2 q' \approx 2\lambda$. Then, concretely for $\lambda = 128$, and operating over fields and elliptic curves of order $2\lambda$, the prover must perform 4096 point multiplications, the verifier must perform 2816 point multiplications, and proof size is roughly 151 kB.

## 6.4 Weak Zero-knowledge in the CROSS Identification Scheme

CROSS [BBB+24] is a candidate submitted to the NIST additional digital signatures competition. The CROSS protocol has been selected as a round 2 candidate and is thus under more scrutiny by the cryptographic community. This is particularly the case for CROSS, which has recently been attacked in [BLP+25], leading the authors to update their parameters to account for the loss in bit-security. In this note, we show that the identification protocol which the CROSS signature is based on does not satisfy their stated definition of honest verifier zero-knowledge. In particular, we show that (a) the distributions of real and simulated transcripts are not statistically close, and (b) given access to the witness, it is possible to distinguish between the real and simulated transcripts with overwhelming advantage.

[6] The Fiat-Shamir transformation should be implemented in the standard manner, such as including all public parameters in the challenge oracle query, in order to prevent any weak Fiat-Shamir attacks [DMWG23].

[7] This is a reasonable assumption, since common parameters, such as the ones for `secp256k1`, implement fields of bit length equal to the elliptic curve group order, and Bröker's algorithm heuristically returns an elliptic curve whose order and base field are the same bit length.

In short, the problem present in CROSS-ID is that the commitments sent to the verifier are not hiding. Generally, a hash-based commitment needs fresh, independent randomness in order to be hiding. We believe this flaw is due to the extreme optimisations of CROSS, in an attempt to minimise their signature sizes.

It is unclear what relevance this has to the security of the resulting signature scheme, when the Fiat-Shamir transform has been applied. However, this is particularly relevant to the security of fully-anonymous ring signatures constructed via generic transformation of identification protocols, as was seen in the case of SQISign [BLL24].

*Note:* We do not include the protocol description of $\Pi_{\mathsf{CROSS\text{-}ID}}$, which is available in the CROSS security document [BBB+25]. The relevant content is presented in Section 4.2 and Figure 4. Hence, except for the definitions of zero-knowledge, we will follow all notation and definitions that are present in the document.

**Variants of Honest Verifier Zero Knowledge** We consider the various notions of honest verifier zero-knowledge in the context of 5-round identification protocols, which are extended from the classic definitions of 3-round sigma protocols. Below is the definition present in the v2.0 security document of CROSS, retrieved from [BBB+25]:

**Definition 17** (Honest Verifier Zero-Knowledge)**.** Let $\Pi = (\mathcal{P}, \mathcal{V})$ be an interactive proof system for a hard relation $\mathcal{R} \subseteq X \times Y$. We say that $\Pi$ is honest-verifier zero-knowledge if there exists a probabilistic polynomial time algorithm $\mathcal{S}$, called the simulator, such that the following two distribution ensembles are indistinguishable:

$$\{(x, w, \langle \mathcal{P}(x, w), \mathcal{V}(x) \rangle) \mid (x, w) \leftarrow\!\$\, \mathcal{R}\} \text{ and } \{(x, w, \mathcal{S}(x)) \mid (x, w) \leftarrow\!\$\, \mathcal{R}\}$$

where $\langle \mathcal{P}(x, w), \mathcal{V}(x) \rangle$ denotes the transcript of an honest interaction between a prover and the verifier with their respective inputs.

There is some ambiguity in the definition. It is unclear what the authors mean by indistinguishability. This may be interpreted as *perfectly* indistinguishable, the distributions are identical; *statistically* indistinguishable, they differ by some negligible statistical distance; or *computationally* indistinguishable, a PPT distinguisher cannot distinguish the distributions with non-negligible advantage. In the security proof for zero-knowledge, they claim that certain values in the simulated transcript follow the same statistical distribution as those in real transcripts, but typically the witness is only included in the distribution given to the distinguisher in the setting of computational zero-knowledge. Unfortunately, we show that the CROSS protocol can only satisfy the variant below (in the ROM):

**Definition 18** (Weak Computational Honest Verifier Zero-knowledge)**.** Given a random oracle $O$, let $\Pi = (\mathcal{P}^O, \mathcal{V}^O)$ be interactive proof system for a hard relation $\mathcal{R} \subseteq X \times Y$.

We say that $\Pi$ is weak computational honest-verifier zero-knowledge if there exists a probabilistic polynomial time algorithm $\mathcal{S}^O$, called the simulator, such that for any $\lambda \in \mathbb{N}$, and any PPT distinguisher $D$ that makes polynomially many queries to the random oracle $O$, the following quantity is negligible in $\lambda$:

$$\left| \Pr[D^O(x, \langle \mathcal{P}^O(x,w), \mathcal{V}^O(x) \rangle) = 1 \mid (x,w) \leftarrow\!\!\$\ \mathcal{R}] - \Pr[D^O(x, \mathcal{S}^O(x)) = 1 \mid (x,w) \leftarrow\!\!\$\ \mathcal{R}] \right|$$

We denote the above quantity as the advantage of $D$.

In particular, we first provide a distinguisher for $\Pi_{\mathsf{CROSS\text{-}ID}}$ which shows that it does not satisfy Definition 17 (i.e. it is not *strong* honest verifier zero-knowledge). Then, we show that $\Pi_{\mathsf{CROSS\text{-}ID}}$ is not statistically zero-knowledge, since for a fixed instance-witness pair, the distributions of real and simulated transcripts are not statistically close.

### 6.4.1 An Efficient Distinguisher Given Access to the Witness

We define a distinguisher $D$ for the distributions from Definition 17 of $\Pi_{\mathsf{CROSS\text{-}ID}}$ as follows. On input:

- instance $(G, \mathbf{H}, \mathbf{s})$ where $G \subseteq \mathbb{E}^n$, $\mathbf{H} \in \mathbb{F}_p^{(n-k)\times n}$, $\mathbf{s} \in \mathbb{F}_p^{n-k}$,
- witness $\mathbf{e} \in G$ such that $\mathbf{s} = \mathbf{e}\mathbf{H}^\top$,
- and transcript $T = (\mathtt{cmt}_0, \mathtt{cmt}_1, \mathtt{chall}_1, \mathtt{digest}_y, \mathtt{chall}_2, \mathtt{resp})$

The distinguisher, given oracle access to $O = \mathsf{Hash}(\cdot)$, performs the following steps:

1. If $\mathtt{chall}_2 = 0$, parse $\mathtt{resp}$ as $\mathsf{Seed}$ and perform the following:
   (a) Compute $(\mathbf{e}', \mathbf{u}') \leftarrow \mathsf{CSPRNG}(\mathsf{Seed})$ and hence with knowledge of $\mathbf{e}$, obtain $\mathbf{v} = \mathbf{e} \star (\mathbf{e}')^{-1}$, and $\mathbf{u} = \mathbf{v} \star \mathbf{u}'$. Lastly, compute $\mathbf{s}' = \mathbf{u}\mathbf{H}^\top$.
   (b) If $\mathtt{cmt}_0 = \mathsf{Hash}(\mathbf{s}' | \mathbf{v})$, output 1. Otherwise, output 0.
2. Otherwise, $\mathtt{chall}_2 = 1$, parse $\mathtt{resp}$ as $(\mathbf{y}, \mathbf{v})$ and perform the following:
   (a) With knowledge of $\mathbf{e}$, compute $\mathbf{e}' = \mathbf{e} \star \mathbf{v}^{-1}$ and solve for $\mathbf{u}' = \mathbf{y} - \mathtt{chall}_1 \star \mathbf{e}'$.
   (b) If $\mathtt{cmt}_1 = \mathsf{Hash}(\mathbf{u}' | \mathbf{v})$, output 1. Otherwise, output 0.

The only time the distinguisher will output 1 for a simulated transcript is when the simulator chooses a random bit string that coincides with the commitment for an honest execution of the protocol with the same challenges and responses. This occurs with probability $2^{-2\lambda}$. Hence the distinguisher has overwhelming advantage $1 - 2^{-2\lambda}$ in distinguishing the distributions in Definition 17.

### 6.4.2 Statistical Distance Between Real and Simulated Transcripts

Recall the statistical distance (or total variable distance) between two distributions (or random variables) $X$ and $Y$ is defined as:

$$\Delta(X,Y) = \frac{1}{2}\sum_x |\Pr[X=x] - \Pr[Y=x]|$$

The distributions are said to be statistically indistinguishable for a parameter $\lambda \in \mathbb{N}$ if $\Delta(X,Y) \leq \mathrm{negl}(\lambda)$.

We rely on the following classical results from [Sho05, Thm 8.32, Thm 8.31].

**Lemma 1.** *If $S$ and $T$ are finite sets, and $X$ and $Y$ are random variables taking values in $S$, and $f : S \to T$ is a function, then $\Delta(X,Y) \geq \Delta(f(X), f(Y))$.*

**Lemma 2.** *Let $X$ and $Y$ be random variables taking the values in a set $S$. For every $S' \subseteq S$, we have $\Delta(X,Y) \geq |\Pr[X \in S'] - \Pr[Y \in S']|$.*

Given $(P^O, V^O) = \Pi_{\mathsf{CROSS\text{-}ID}}$, let us consider the distribution of transcripts for a fixed $(x,w) \in \mathcal{R}_{\mathsf{CROSS\text{-}ID}}$ and instantiation of a random oracle $O$. Let

$$\mathcal{T}_{\text{real}} = \{\langle P^O(x,w), V^O(x)\rangle\} \text{ and } \mathcal{T}_{\text{sim}} = \{\mathcal{S}^O(x)\}$$

be the random variables associated to real and simulated transcripts respectively over the set of valid transcripts $\Omega$. From now on, parse elements of the set $T \in \Omega$ as

$$(\mathtt{cmt}_0, \mathtt{cmt}_1, \mathtt{chall}_1, \mathtt{digest}_y, \mathtt{chall}_2, \mathtt{resp}).$$

First, we observe that the real transcripts use a $\lambda$-bit seed, which determines the resulting values for $\mathtt{cmt}_0$ and $\mathtt{cmt}_1$. However, the digests for the random oracle are length $2\lambda$. Hence there are at least $2^{2\lambda} - 2^\lambda$ possible values for $\mathtt{cmt}_0$ and $\mathtt{cmt}_1 \in \{0,1\}^{2\lambda}$ which are never used in the real transcripts, but are in the unopened commitment of the simulated transcript. Consider the function $f : \Omega \to \{0,1\} \times \{0,1\}^{2\lambda}$ which sends:

$$(\mathtt{cmt}_0, \mathtt{cmt}_1, \mathtt{chall}_1, \mathtt{digest}_y, \mathtt{chall}_2, \mathtt{resp}) \mapsto (\mathtt{chall}_2, \mathtt{cmt}_{1-\mathtt{chall}_2})$$

That is, the distribution of $\mathtt{chall}_2$ and the $(1-\mathtt{chall}_2)$-th commitment, which remains unopened. Let $\mathcal{T}'_{\text{real}} := f(\mathcal{T}_{\text{real}})$ and $\mathcal{T}'_{\text{sim}} := f(\mathcal{T}_{\text{sim}})$, and observe that the latter distribution is uniformly distributed.

Define the function $g_{0,\mathbf{e}} : \{0,1\}^\lambda \to \{0,1\}^{2\lambda}$ that takes as input a $\lambda$-bit seed, and outputs the resulting commitment $\mathtt{cmt}_0$ in the real transcript for a given witness $\mathbf{e}$. Similarly, define $g_{1,\mathbf{e}}$ to output the commitment $\mathtt{cmt}_1$ in the real transcript for a given seed and $\mathbf{e}$. Now, let

$$S_0 = \{0\} \times \{0,1\}^{2\lambda} \setminus \{0\} \times g_{1,\mathbf{e}}(\{0,1\}^\lambda),$$
$$\text{and } S_1 = \{1\} \times \{0,1\}^{2\lambda} \setminus \{1\} \times g_{0,\mathbf{e}}(\{0,1\}^\lambda)$$

Which is precisely the set of values which arise in $\mathcal{T}'_{\text{sim}}$ and not in $\mathcal{T}'_{\text{real}}$. We note that

$$|S_0| \geq 2^{2\lambda} - 2^{\lambda} \text{ and } |S_1| \geq 2^{2\lambda} - 2^{\lambda}. \tag{6.5}$$

Equality is the best case, where $g_{i,\mathbf{e}}$ is injective for both $i \in \{0,1\}$. Hence, we have:

$$\begin{aligned}
\Delta(\mathcal{T}_{\text{real}}, \mathcal{T}_{\text{sim}}) &\geq \Delta(\mathcal{T}'_{\text{real}}, \mathcal{T}'_{\text{sim}}) && \text{(By Lemma 1)} \\
&\geq |\Pr[\mathcal{T}'_{\text{real}} \in S_0 \cup S_1] - \Pr[\mathcal{T}'_{\text{sim}} \in S_0 \cup S_1]| && \text{(By Lemma 2)} \\
&= |0 - \Pr[\mathcal{T}'_{\text{sim}} \in S_0 \cup S_1]| \\
&= \Pr[\mathcal{T}'_{\text{sim}} \in S_0 \cup S_1] \\
&= \frac{|S_0 \cup S_1|}{|\{0,1\} \times \{0,1\}^{2\lambda}|} \\
&\geq \frac{2^{2\lambda} - 2^{\lambda} + 2^{2\lambda} - 2^{\lambda}}{2 \cdot 2^{2\lambda}} && \text{(By Equation (6.5))} \\
&= 1 - 2^{-\lambda}
\end{aligned}$$

Hence we have that real and simulated protocol transcripts for a fixed instance $(x, w) \in \mathcal{R}_{\mathsf{CROSS\text{-}ID}}$ are far from being statistically indistinguishable, with statistical distance at least $1 - 2^{-\lambda} = 1 - \text{negl}(\lambda)$.

# Bibliography


[ABCP23] Shahla Atapoor, Karim Baghery, Daniele Cozzo, and Robi Pedersen. CSI-SharK: CSI-FiSh with sharing-friendly keys. In Leonie Simpson and Mir Ali Rezazadeh Baee, editors, *ACISP 23*, volume 13915 of *LNCS*, pages 471–502. Springer, Cham, July 2023.

[ACFY25] Gal Arnon, Alessandro Chiesa, Giacomo Fenzi, and Eylon Yogev. WHIR: Reed-solomon proximity testing with super-fast verification. In Serge Fehr and Pierre-Alain Fouque, editors, *EUROCRYPT 2025, Part IV*, volume 15604 of *LNCS*, pages 214–243. Springer, Cham, May 2025.

[ACNL+21] Sarah Arpin, Catalina Camacho-Navarro, Kristin Lauter, Joelle Lim, Kristina Nelson, Travis Scholl, and Jana Sotáková and. Adventures in supersingularland. *Experimental Mathematics*, 32(2), 2021.

[ADMP20] Navid Alamati, Luca De Feo, Hart Montgomery, and Sikhar Patranabis. Cryptographic group actions and applications. In Shiho Moriai and Huaxiong Wang, editors, *ASIACRYPT 2020, Part II*, volume 12492 of *LNCS*, pages 411–439. Springer, Cham, December 2020.

[AEK+22] Michel Abdalla, Thorsten Eisenhofer, Eike Kiltz, Sabrina Kunzweiler, and Doreen Riepel. Password-authenticated key exchange from group actions. In Yevgeniy Dodis and Thomas Shrimpton, editors, *CRYPTO 2022, Part II*, volume 13508 of *LNCS*, pages 699–728. Springer, Cham, August 2022.

[AFK22] Thomas Attema, Serge Fehr, and Michael Klooß. Fiat-shamir transformation of multi-round interactive proofs. In Eike Kiltz and Vinod Vaikuntanathan, editors, *TCC 2022, Part I*, volume 13747 of *LNCS*, pages 113–142. Springer, Cham, November 2022.

[AFK23] Thomas Attema, Serge Fehr, and Michael Klooß. Fiat-shamir transformation of multi-round interactive proofs (extended version). *Journal of Cryptology*, 36(4):36, October 2023.

[AGM18] Shashank Agrawal, Chaya Ganesh, and Payman Mohassel. Non-interactive zero-knowledge proofs for composite statements. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part III*, volume 10993 of *LNCS*, pages 643–673. Springer, Cham, August 2018.

[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.

[AIL+21] Laia Amorós, Annamaria Iezzi, Kristin Lauter, Chloe Martindale, and Jana Sotáková. *Explicit Connections Between Supersingular Isogeny Graphs and Bruhat–Tits Trees*, pages 39–73. Springer International Publishing, Cham, 2021.

[AOS02] Masayuki Abe, Miyako Ohkubo, and Koutarou Suzuki. 1-out-of-n signatures from a variety of keys. In Yuliang Zheng, editor, *ASIACRYPT 2002*, volume 2501 of *LNCS*, pages 415–432. Springer, Berlin, Heidelberg, December 2002.

[Bas24a] Andrea Basso. POKE: A framework for efficient PKEs, split KEMs, and OPRFs from higher-dimensional isogenies. Cryptology ePrint Archive, Report 2024/624, 2024.

[Bas24b] Andrea Basso. A post-quantum round-optimal oblivious PRF from isogenies. In Claude Carlet, Kalikinkar Mandal, and Vincent Rijmen, editors, *SAC 2023*, volume 14201 of *LNCS*, pages 147–168. Springer, Cham, August 2024.

[BBB+18] Benedikt Bünz, Jonathan Bootle, Dan Boneh, Andrew Poelstra, Pieter Wuille, and Greg Maxwell. Bulletproofs: Short proofs for confidential transactions and more. In *2018 IEEE Symposium on Security and Privacy*, pages 315–334. IEEE Computer Society Press, May 2018.

[BBB+24] Marco Baldi, Alessandro Barenghi, Michele Battagliola, Sebastian Bitzer, Marco Gianvecchio, Patrick Karl, Felice Manganiello, Alessio Pavoni, Gerardo Pelosi, Paolo Santini, Jonas Schupp, Edoardo Signorini, Freeman Slaughter, Antonia Wachter-Zeh, and Violetta Weger. CROSS — Codes and Restricted Objects Signature Scheme. Technical report, National Institute of Standards and Technology, 2024. available at https://csrc.nist.gov/Projects/pqc-dig-sig/round-2-additional-signatures.

[BBB+25] Marco Baldi, Alessandro Barenghi, Sebastian Bitzer, Patrick Karl, Felice Manganiello, Alessio Pavoni, Gerardo Pelosi, Paolo Santini, Jonas Schupp,

Freeman Slaughter, Antonia Wachter-Zeh, and Violetta Weger. CROSS security details document v2.0, 2025.

[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.

[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+23] Andrea Basso, Giulio Codogni, Deirdre Connolly, Luca De Feo, Tako Boris Fouotsa, Guido Maria Lido, Travis Morrison, Lorenz Panny, Sikhar Patranabis, and Benjamin Wesolowski. Supersingular curves you can trust. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part II*, volume 14005 of *LNCS*, pages 405–437. Springer, Cham, April 2023.

[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.

[BCG+17] Jonathan Bootle, Andrea Cerulli, Essam Ghadafi, Jens Groth, Mohammad Hajiabadi, and Sune K. Jakobsen. Linear-time zero-knowledge proofs for arithmetic circuit satisfiability. In Tsuyoshi Takagi and Thomas Peyrin, editors, *ASIACRYPT 2017, Part III*, volume 10626 of *LNCS*, pages 336–365. Springer, Cham, December 2017.

[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, Cham, 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, Berlin, Heidelberg, October / November 2016.

[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, Cham, October 2021.

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

[BDE+22] Maxime Buser, Rafael Dowsley, Muhammed F. Esgin, Shabnam Kasra Kermanshahi, Veronika Kuchta, Joseph K. Liu, Raphaël C.-W. Phan, and Zhenfei Zhang. Post-quantum verifiable random function from symmetric primitives in PoS blockchain. In Vijayalakshmi Atluri, Roberto Di Pietro, Christian Damsgaard Jensen, and Weizhi Meng, editors, *ESORICS 2022, Part I*, volume 13554 of *LNCS*, pages 25–45. Springer, Cham, September 2022.

[BDK+21] Carsten Baum, Cyprien Delpech 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, Cham, 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, Cham, May / June 2022.

[BFK+24] Alexander R. Block, Zhiyong Fang, Jonathan Katz, Justin Thaler, Hendrik Waldner, and Yupeng Zhang. Field-agnostic SNARKs from expand-accumulate codes. In Leonid Reyzin and Douglas Stebila, editors, *CRYPTO 2024, Part X*, volume 14929 of *LNCS*, pages 276–307. Springer, Cham, August 2024.

[BG93] Mihir Bellare and Oded Goldreich. On defining proofs of knowledge. In Ernest F. Brickell, editor, *Advances in Cryptology — CRYPTO' 92*, pages 390–420, Berlin, Heidelberg, 1993. Springer Berlin Heidelberg.

[BJS14] Jean-François Biasse, David Jao, and Anirudh Sankar. A quantum algorithm for computing isogenies between supersingular elliptic curves. In Willi Meier and Debdeep Mukhopadhyay, editors, *INDOCRYPT 2014*, volume 8885 of *LNCS*, pages 428–442. Springer, Cham, December 2014.

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

[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, Cham, 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, Cham, December 2019.

[BKW20] Dan Boneh, Dmitry Kogan, and Katharine Woo. Oblivious pseudorandom functions from isogenies. In Shiho Moriai and Huaxiong Wang, editors, *ASIACRYPT 2020, Part II*, volume 12492 of *LNCS*, pages 520–550. Springer, Cham, December 2020.

[BLL24] Giacomo Borin, Yi-Fu Lai, and Antonin Leroux. Erebor and durian: Full anonymous ring signatures from quaternions and isogenies. *CiC*, 1(4):4, 2024.

[BLP+25] Michele Battagliola, Riccardo Longo, Federico Pintore, Edoardo Signorini, and Giovanni Tognolini. A revision of CROSS security: Proofs and attacks for multi-round fiat-shamir signatures. Cryptology ePrint Archive, Paper 2025/127, 2025.

[BMP23] Andrea Basso, Luciano Maino, and Giacomo Pope. FESTA: Fast encryption from supersingular torsion attacks. In Jian Guo and Ron Steinfeld, editors, *ASIACRYPT 2023, Part VII*, volume 14444 of *LNCS*, pages 98–126. Springer, Singapore, December 2023.

[BNPS03] Mihir Bellare, Chanathip Namprempre, David Pointcheval, and Michael Semanko. The one-more-RSA-inversion problems and the security of Chaum's blind signature scheme. *Journal of Cryptology*, 16(3):185–215, June 2003.

[Bou00] Fabrice Boudot. Efficient proofs that a committed number lies in an interval. In Bart Preneel, editor, *EUROCRYPT 2000*, volume 1807 of *LNCS*, pages 431–444. Springer, Berlin, Heidelberg, May 2000.

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In Dorothy E. Denning, Raymond Pyle, Ravi Ganesan, Ravi S. Sandhu, and Victoria Ashby, editors, *ACM CCS 93*, pages 62–73. ACM Press, November 1993.

[Bra97] Stefan Brands. Rapid demonstration of linear relations connected by Boolean operators. In Walter Fumy, editor, *EUROCRYPT'97*, volume 1233 of *LNCS*, pages 318–333. Springer, Berlin, Heidelberg, May 1997.

[BS07] Reinier Broker and Peter Stevenhagen. Constructing elliptic curves of prime order, 2007.

[BSN24] Omid Bodaghi and Reihaneh Safavi-Naini. Short Paper: Breaking X-VRF, a Post-Quantum Verifiable Random Function. In *Financial Crypto 2024*, March 2024. Available at https://fc24.ifca.ai/preproceedings/213.pdf.

[CD20] Wouter Castryck and Thomas Decru. CSIDH on the surface. In Jintai Ding and Jean-Pierre Tillich, editors, *Post-Quantum Cryptography - 11th International Conference, PQCrypto 2020*, pages 111–129. Springer, Cham, 2020.

[CD23] Wouter Castryck and Thomas Decru. An efficient key recovery attack on SIDH. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part V*, volume 14008 of *LNCS*, pages 423–447. Springer, Cham, April 2023.

[CDH+25] Sofía Celi, Alex Davidson, Hamed Haddadi, Gonçalo Pestana, and Joe Rowell. DiStefano: Decentralized infrastructure for sharing trusted encrypted facts and nothing more. In *NDSS 2025*. The Internet Society, February 2025.

[CDS94] Ronald Cramer, Ivan Damgård, and Berry Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In Yvo Desmedt, editor, *CRYPTO'94*, volume 839 of *LNCS*, pages 174–187. Springer, Berlin, Heidelberg, August 1994.

[CDV20] Wouter Castryck, Thomas Decru, and Frederik Vercauteren. Radical isogenies. In Shiho Moriai and Huaxiong Wang, editors, *ASIACRYPT 2020, Part II*, volume 12492 of *LNCS*, pages 493–519. Springer, Cham, December 2020.

[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.

[CLL23] Kelong Cong, Yi-Fu Lai, and Shai Levin. Efficient isogeny proofs using generic techniques. In Mehdi Tibouchi and Xiaofeng Wang, editors, *ACNS 2023, Part II*, volume 13906 of *LNCS*, pages 248–275. Springer, Cham, June 2023.

[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, Cham, December 2018.

[CLR24] Sofía Celi, Shai Levin, and Joe Rowell. CDLS: Proving knowledge of committed discrete logarithms with soundness. In Serge Vaudenay and Christophe Petit, editors, *AFRICACRYPT 24*, volume 14861 of *LNCS*, pages 69–93. Springer, Cham, July 2024.

[CM99] Jan Camenisch and Markus Michels. Proving in zero-knowledge that a number is the product of two safe primes. In Jacques Stern, editor, *EUROCRYPT'99*, volume 1592 of *LNCS*, pages 107–122. Springer, Berlin, Heidelberg, May 1999.

[CM19] Jing Chen and Silvio Micali. Algorand: A secure and efficient distributed ledger. *Theor. Comput. Sci.*, 777:155–183, 2019.

[CMB23] Kevin Choi, Aathira Manoj, and Joseph Bonneau. SoK: Distributed randomness beacons. In *2023 IEEE Symposium on Security and Privacy*, pages 75–92. IEEE Computer Society Press, May 2023.

[Cou06] Jean-Marc Couveignes. Hard homogeneous spaces. Cryptology ePrint Archive, Report 2006/291, 2006.

[CP93] David Chaum and Torben P. Pedersen. Wallet databases with observers. In Ernest F. Brickell, editor, *CRYPTO'92*, volume 740 of *LNCS*, pages 89–105. Springer, Berlin, Heidelberg, August 1993.

[CS97] Jan Camenisch and Markus Stadler. Efficient group signature schemes for large groups (extended abstract). In Burton S. Kaliski, Jr., editor, *CRYPTO'97*, volume 1294 of *LNCS*, pages 410–424. Springer, Berlin, Heidelberg, August 1997.

[CSRT22] Jorge Chávez-Saab, Francisco Rodríguez-Henríquez, and Mehdi Tibouchi. Verifiable isogeny walks: Towards an isogeny-based postquantum VDF. In Riham AlTawy and Andreas Hülsing, editors, *SAC 2021*, volume 13203 of *LNCS*, pages 441–460. Springer, Cham, September / October 2022.

[Dam10] Ivan Damgård. On sigma-protocols, 2010. https://www.cs.au.dk/~ivan/Sigma.pdf.

[DDGZ22] Luca De Feo, Samuel Dobson, Steven D. Galbraith, and Lukas Zobernig. SIDH proof of knowledge. In Shweta Agrawal and Dongdai Lin, editors, *ASIACRYPT 2022, Part II*, volume 13792 of *LNCS*, pages 310–339. Springer, Cham, December 2022.

[DDOS19] Cyprien Delpech 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, Cham, August 2019.

[DF24] Max Duparc and Tako Boris Fouotsa. SQIPrime: A dimension 2 variant of SQISignHD with non-smooth challenge isogenies. In Kai-Min Chung and Yu Sasaki, editors, *ASIACRYPT 2024, Part III*, volume 15486 of *LNCS*, pages 396–429. Springer, Singapore, December 2024.

[DG16] Christina Delfs and Steven D. Galbraith. Computing isogenies between supersingular elliptic curves over $\mathbb{F}_p$. *DCC*, 78(2):425–440, 2016.

[DG19] Luca De Feo and Steven D. Galbraith. SeaSign: Compact isogeny signatures from class group actions. In Yuval Ishai and Vincent Rijmen, editors, *EUROCRYPT 2019, Part III*, volume 11478 of *LNCS*, pages 759–789. Springer, Cham, May 2019.

[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, Cham, December 2020.

[DLRW24] Pierrick Dartois, Antonin Leroux, Damien Robert, and Benjamin Wesolowski. SQIsignHD: New dimensions in cryptography. In Marc Joye and Gregor Leander, editors, *EUROCRYPT 2024, Part I*, volume 14651 of *LNCS*, pages 3–32. Springer, Cham, May 2024.

[DMPS19] Luca De Feo, Simon Masson, Christophe Petit, and Antonio Sanso. Verifiable delay functions from supersingular isogenies and pairings. In Steven D. Galbraith and Shiho Moriai, editors, *ASIACRYPT 2019, Part I*, volume 11921 of *LNCS*, pages 248–277. Springer, Cham, December 2019.

[DMWG23] Quang Dao, Jim Miller, Opal Wright, and Paul Grubbs. Weak fiat-shamir attacks on modern proof systems. In *2023 IEEE Symposium on Security and Privacy*, pages 199–216. IEEE Computer Society Press, May 2023.

[DP24] Benjamin E. Diamond and Jim Posen. Proximity testing with logarithmic randomness. *CiC*, 1(1):2, 2024.

[DPB24] Javad Doliskani, Geovandro C. C. F. Pereira, and Paulo S. L. M. Barreto. Faster cryptographic hash function from supersingular isogeny graphs. In Benjamin Smith and Huapeng Wu, editors, *SAC 2022*, volume 13742 of *LNCS*, pages 399–415. Springer, Cham, August 2024.

[DPV19] Thomas Decru, Lorenz Panny, and Frederik Vercauteren. Faster SeaSign signatures through improved rejection sampling. In Jintai Ding and Rainer Steinwandt, editors, *Post-Quantum Cryptography - 10th International Conference, PQCrypto 2019*, pages 271–285. Springer, Cham, 2019.

[EEK+23] Muhammed F. Esgin, Oguzhan Ersoy, Veronika Kuchta, Julian Loss, Amin Sakzad, Ron Steinfeld, Xiangwen Yang, and Raymond K. Zhao. A new look at blockchain leader election: Simple, efficient, sustainable and post-quantum. In Joseph K. Liu, Yang Xiang, Surya Nepal, and Gene Tsudik, editors, *Proceedings of the 2023 ACM Asia Conference on Computer and Communications Security, ASIA CCS 2023, Melbourne, VIC, Australia, July 10-14, 2023*, pages 623–637. ACM, 2023.

[EKS+21] Muhammed F. Esgin, Veronika Kuchta, Amin Sakzad, Ron Steinfeld, Zhenfei Zhang, Shifeng Sun, and Shumo Chu. Practical post-quantum few-time verifiable random function with applications to algorand. In Nikita Borisov and Claudia Díaz, editors, *FC 2021, Part II*, volume 12675 of *LNCS*, pages 560–578. Springer, Berlin, Heidelberg, March 2021.

[ESLR23] Muhammed F. Esgin, Ron Steinfeld, Dongxi Liu, and Sushmita Ruj. Efficient hybrid exact/relaxed lattice proofs and applications to rounding and VRFs. In Helena Handschuh and Anna Lysyanskaya, editors, *CRYPTO 2023, Part V*, volume 14085 of *LNCS*, pages 484–517. Springer, Cham, August 2023.

[FFK+23] Antonio Faonio, Dario Fiore, Markulf Kohlweiss, Luigi Russo, and Michal Zajac. From polynomial IOP and commitments to non-malleable zk-SNARKs. In Guy N. Rothblum and Hoeteck Wee, editors, *TCC 2023, Part III*, volume 14371 of *LNCS*, pages 455–485. Springer, Cham, November / December 2023.

[FLM22] Armando Faz-Hernández, Watson Ladd, and Deepak Maram. ZKAttest: Ring and group signatures for existing ECDSA keys. In Riham AlTawy and Andreas Hülsing, editors, *SAC 2021*, volume 13203 of *LNCS*, pages 68–83. Springer, Cham, September / October 2022.

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

[FO97] Eiichiro Fujisaki and Tatsuaki Okamoto. Statistical zero knowledge protocols to prove modular polynomial relations. In Burton S. Kaliski, Jr., editor, *CRYPTO'97*, volume 1294 of *LNCS*, pages 16–30. Springer, Berlin, Heidelberg, August 1997.

[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, Berlin, Heidelberg, August 1987.

[GGLM24] Steven Galbraith, Valerie Gilchrist, Shai Levin, and Ari Markowitz. Further connections between isogenies of supersingular curves and bruhat-tits trees. Cryptology ePrint Archive, Paper 2024/1971, 2024.

[GK15] Jens Groth and Markulf Kohlweiss. One-out-of-many proofs: Or how to leak a secret and spend a coin. In Elisabeth Oswald and Marc Fischlin, editors, *EUROCRYPT 2015, Part II*, volume 9057 of *LNCS*, pages 253–280. Springer, Berlin, Heidelberg, April 2015.

[GLH+25] Yanpei Guo, Xuanming Liu, Kexi Huang, Wenjie Qu, Tianyang Tao, and Jiaheng Zhang. DeepFold: Efficient multilinear polynomial commitment from reed-solomon code and its application to zero-knowledge proofs. In Lujo Bauer and Giancarlo Pellegrino, editors, *USENIX Security 2025*, pages 3497–3516. USENIX Association, August 2025.

[GLS+23] Alexander Golovnev, Jonathan Lee, Srinath T. V. Setty, Justin Thaler, and Riad S. Wahby. Brakedown: Linear-time and field-agnostic SNARKs

for R1CS. In Helena Handschuh and Anna Lysyanskaya, editors, *CRYPTO 2023, Part II*, volume 14082 of *LNCS*, pages 193–226. Springer, Cham, August 2023.

[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.

[GMR89] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof systems. *SIAM Journal on Computing*, 18(1):186–208, 1989.

[GNP+15] Sharon Goldberg, Moni Naor, Dimitrios Papadopoulos, Leonid Reyzin, Sachin Vasant, and Asaf Ziv. NSEC5: Provably preventing DNSSEC zone enumeration. In *NDSS 2015*. The Internet Society, February 2015.

[GOP+25] Chaya Ganesh, Claudio Orlandi, Mahak Pancholi, Akira Takahashi, and Daniel Tschudi. Fiat-shamir bulletproofs are non-malleable (in the random oracle model). *Journal of Cryptology*, 38(1):11, January 2025.

[GOT19] Chaya Ganesh, Claudio Orlandi, and Daniel Tschudi. Proof-of-stake protocols for privacy-aware blockchains. In Yuval Ishai and Vincent Rijmen, editors, *EUROCRYPT 2019, Part I*, volume 11476 of *LNCS*, pages 690–719. Springer, Cham, May 2019.

[GPS17] Steven D. Galbraith, Christophe Petit, and Javier Silva. Identification protocols and signature schemes based on supersingular isogeny problems. In Tsuyoshi Takagi and Thomas Peyrin, editors, *ASIACRYPT 2017, Part I*, volume 10624 of *LNCS*, pages 3–33. Springer, Cham, December 2017.

[GPST16] Steven D. Galbraith, Christophe Petit, Barak Shani, and Yan Bo Ti. On the security of supersingular isogeny cryptosystems. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *ASIACRYPT 2016, Part I*, volume 10031 of *LNCS*, pages 63–91. Springer, Berlin, Heidelberg, December 2016.

[GS24] Emanuele Giunta and Alistair Stewart. Unbiasable verifiable random functions. In Marc Joye and Gregor Leander, editors, *EUROCRYPT 2024, Part IV*, volume 14654 of *LNCS*, pages 142–167. Springer, Cham, May 2024.

[Hab23] Ulrich Haböck. Brakedown's expander code. Cryptology ePrint Archive, Report 2023/769, 2023.

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

[Kim24] Suhri Kim. Optimized SeaSign for Enhanced Efficiency. *IEEE Access*, pages 1–1, 2024.

[KLPT14] David Kohel, Kristin E. Lauter, Christophe Petit, and Jean-Pierre Tignol. On the quaternion $\ell$-isogeny path problem. *LMS Journal of Computation and Mathematics*, 17:418–432, 2014.

[KPT23] Markulf Kohlweiss, Mahak Pancholi, and Akira Takahashi. How to compile polynomial IOP into simulation-extractable SNARKs: A modular approach. In Guy N. Rothblum and Hoeteck Wee, editors, *TCC 2023, Part III*, volume 14371 of *LNCS*, pages 486–512. Springer, Cham, November / December 2023.

[Lai24] Yi-Fu Lai. Capybara and tsubaki: Verifiable random functions from group actions and isogenies. *CiC*, 1(3):1, 2024.

[Ler25] Antonin Leroux. Verifiable random function from the deuring correspondence and higher dimensional isogenies. In Serge Fehr and Pierre-Alain Fouque, editors, *EUROCRYPT 2025, Part VII*, volume 15607 of *LNCS*, pages 167–194. Springer, Cham, May 2025.

[Lev25a] Shai Levin. A key-recovery attack on a leaky seasign variant. *IACR Communications in Cryptology*, 1(4), 2025.

[Lev25b] Shai Levin. A note on zero-knowledge simulator of the CROSS identification protocol. Cryptology ePrint Archive, Paper 2025/359, 2025.

[LGD21] Yi-Fu Lai, Steven D. Galbraith, and Cyprien Delpech 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, Cham, October 2021.

[LP25] Shai Levin and Robi Pedersen. Faster proofs and vrfs from isogenies. In Goichiro Hanaoka and Bo-Yin Yang, editors, *Advances in Cryptology – ASIACRYPT 2025*, pages 307–340, Singapore, 2025. Springer Nature Singapore.

[Lyu09] Vadim Lyubashevsky. Fiat-Shamir with aborts: Applications to lattice and factoring-based signatures. In Mitsuru Matsui, editor, *ASIACRYPT 2009*, volume 5912 of *LNCS*, pages 598–616. Springer, Berlin, Heidelberg, December 2009.

[MMP22] Marzio Mula, Nadir Murru, and Federico Pintore. Random sampling of supersingular elliptic curves. Cryptology ePrint Archive, Report 2022/528, 2022.

[MMP+23] Luciano Maino, Chloe Martindale, Lorenz Panny, Giacomo Pope, and Benjamin Wesolowski. A direct key recovery attack on SIDH. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part V*, volume 14008 of *LNCS*, pages 448–471. Springer, Cham, April 2023.

[MRV99] Silvio Micali, Michael O. Rabin, and Salil P. Vadhan. Verifiable random functions. In *40th FOCS*, pages 120–130. IEEE Computer Society Press, October 1999.

[NBMV99] Khanh Quoc Nguyen, Feng Bao, Yi Mu, and Vijay Varadharajan. Zero-knowledge proofs of possession of digital signatures and its applications. In Vijay Varadharajan and Yi Mu, editors, *ICICS 99*, volume 1726 of *LNCS*, pages 103–118. Springer, Berlin, Heidelberg, November 1999.

[NOC+24] Kohei Nakagawa, Hiroshi Onuki, Wouter Castryck, Mingjie Chen, Riccardo Invernizzi, Gioella Lorenzon, and Frederik Vercauteren. SQIsign2D-East: A new signature scheme using 2-dimensional isogenies. In Kai-Min Chung and Yu Sasaki, editors, *ASIACRYPT 2024, Part III*, volume 15486 of *LNCS*, pages 272–303. Springer, Singapore, December 2024.

[NR99] Moni Naor and Omer Reingold. Synthesizers and their application to the parallel construction of pseudo-random functions. *J. Comput. Syst. Sci.*, 58(2):336–375, 1999.

[Ped92] Torben Pryds Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In Joan Feigenbaum, editor, *Advances in Cryptology — CRYPTO '91*, pages 129–140, Berlin, Heidelberg, 1992. Springer Berlin Heidelberg.

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

[PL17] Christophe Petit and Kristin Lauter. Hard and easy problems for supersingular isogeny graphs. Cryptology ePrint Archive, Report 2017/962, 2017.

[Rob23] Damien Robert. Breaking SIDH in polynomial time. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part V*, volume 14008 of *LNCS*, pages 472–503. Springer, Cham, April 2023.

[Sch91] Claus-Peter Schnorr. Efficient signature generation by smart cards. *Journal of Cryptology*, 4(3):161–174, January 1991.

[Sho97] Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal on Computing*, 26(5):1484–1509, 1997.

[Sho05] Victor Shoup. *A Computational Introduction to Number Theory and Algebra*. Cambridge University Press, 2005.

[Sil86] Joseph H. Silverman. *The arithmetic of elliptic curves*, volume 106 of *Graduate texts in mathematics*. Springer, 2nd edition, 1986.

[Ste22] Bruno Sterner. Commitment schemes from supersingular elliptic curve isogeny graphs. 1(2):40–51, 2022.

[Tan09] Seiichiro Tani. Claw finding algorithms using quantum walk. 410(50):5285–5297, 2009.

[Tha23] Justin Thaler. Proofs, arguments, and zero-knowledge. Georgetown University Website, 2023.

[Unr17] Dominique Unruh. Post-quantum security of Fiat-Shamir. In Tsuyoshi Takagi and Thomas Peyrin, editors, *ASIACRYPT 2017, Part I*, volume 10624 of *LNCS*, pages 65–95. Springer, Cham, December 2017.

[Vél71] Jacques Vélu. Isogénies entre courbes elliptiques. *Comptes-Rendus de l'Académie des Sciences*, 273:238–241, 1971.

[WTs+18] Riad S. Wahby, Ioanna Tzialla, abhi shelat, Justin Thaler, and Michael Walfish. Doubly-efficient zkSNARKs without trusted setup. In *2018 IEEE Symposium on Security and Privacy*, pages 926–943. IEEE Computer Society Press, May 2018.

[XZS22] Tiancheng Xie, Yupeng Zhang, and Dawn Song. Orion: Zero knowledge proof with linear prover time. In Yevgeniy Dodis and Thomas Shrimpton, editors, *CRYPTO 2022, Part IV*, volume 13510 of *LNCS*, pages 299–328. Springer, Cham, August 2022.

[XZZ+19] Tiancheng Xie, Jiaheng Zhang, Yupeng Zhang, Charalampos Papamanthou, and Dawn Song. Libra: Succinct zero-knowledge proofs with optimal prover computation. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part III*, volume 11694 of *LNCS*, pages 733–764. Springer, Cham, August 2019.

[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/post-quantum-cryptography-standardization/round-3-submissions.

[ZCF24] Hadas Zeilberger, Binyi Chen, and Ben Fisch. BaseFold: Efficient field-agnostic polynomial commitment schemes from foldable codes. In Leonid Reyzin and Douglas Stebila, editors, *CRYPTO 2024, Part X*, volume 14929 of *LNCS*, pages 138–169. Springer, Cham, August 2024.

[Zha09] Shengyu Zhang. Promised and distributed quantum search. In Lusheng Wang, editor, *Computing and Combinatorics*, pages 430–439. Springer Berlin Heidelberg, 2009.

[ZSP+18] Gustavo Zanon, Marcos A. Simplício, Jr., Geovandro C. C. F. Pereira, Javad Doliskani, and Paulo S. L. M. Barreto. Faster isogeny-based compressed key agreement. In Tanja Lange and Rainer Steinwandt, editors, *Post-Quantum Cryptography - 9th International Conference, PQCrypto 2018*, pages 248–268. Springer, Cham, 2018.