

PRISM with a pinch of salt: Simple, Efficient and Strongly Unforgeable Signatures from Isogenies

Andrea Basso¹, Giacomo Borin^{1,2}, Wouter Castryck³, Maria Corte-Real Santos⁴,
Riccardo Invernizzi³, Antonin Leroux^{5,6}, Luciano Maino^{7,8,9}, Frederik
Vercauteren³, and Benjamin Wesolowski⁴

¹ IBM Research Europe, Zürich, Switzerland

² University of Zürich, Switzerland

³ COSIC, KU Leuven, Belgium

⁴ ENS de Lyon, CNRS, UMPA, UMR 5669, Lyon, France

⁵ DGA-MI, Bruz, France

⁶ IRMAR - UMR 6625, Université de Rennes, France

⁷ University of Birmingham, Birmingham, UK

⁸ Université Libre de Bruxelles, Belgium

⁹ University of Bristol, Bristol, UK

Abstract. The problem of computing an isogeny of large prime degree from a supersingular elliptic curve of unknown endomorphism ring is assumed to be hard both for classical as well as quantum computers. In this work, we first build a two-round identification protocol whose security reduces to this problem. The challenge consists of a random large prime q and the prover simply replies with an efficient representation of an isogeny of degree q from its public key. Using the hash-and-sign paradigm, we then derive a signature scheme with a very simple and flexible signing procedure and prove its security in the standard model. The most efficient variant of our signature schemes features a signing which is $1.4\times$ to $1.6\times$ faster than the most recent implementation of SQIsign, whereas verification ranges from $1.2\times$ slower to $1.01\times$ faster depending on the security level. The sizes of public key and signature are comparable to existing schemes.

Keywords: Isogenies · Post-Quantum · Signatures · ID Protocols

This article is an extended version of [7] in which we additionally present two variants of the PRISM signature: one that achieves strong unforgeability, and another called **salt-PRISM** that allows for smaller parameters. We also comment on the attack from [53]. We provide a C implementation of **salt-PRISM** based on the round 2 NIST submission of SQIsign [1], and compare it to the state-of-the-art implementations of isogeny-based signature schemes. Most notably, at Level V, **salt-PRISM** is the fastest isogeny-based signature to date.

PRISM stands for PRime degree ISogeny Mechanism

1 Introduction

Post-quantum cryptography aims to construct cryptographic protocols that are secure against adversaries with access to both classical and quantum computers. This field has recently gained interest from the community due to the increased investment into quantum computing. Most notably, in 2016, NIST began an effort to standardise post-quantum secure key encapsulation mechanisms (KEMs) and digital signature schemes. This culminated in the standardisation of the lattice-based KEM Kyber [18] and signature schemes Dilithium [33], Falcon [38], and the hash-based signature SPHINCS+ [12]. NIST, however, is still seeking alternative signature schemes for standardisation [56] due to the heavy reliance on lattice-based assumptions, and the relatively large signature sizes compared to pre-quantum alternatives.

Isogeny-based cryptography offers a promising answer to these problems. For instance, the isogeny-based signature scheme SQIsign boasts the smallest combined public key and signature size of any post-quantum alternative. It is currently submitted to Round 2 of NIST’s alternate call for post-quantum secure signature schemes [1]. The main disadvantage of SQIsign and isogeny-based schemes in general is their inefficiency: signing and verification are orders of magnitude slower than lattice-based alternatives. As such, a large portion of research in isogenies has focused on optimizing the subroutines of SQIsign, see for example [28,21]. Recently, impressive strides have been made in this regard by exploiting the attacks on SIDH/SIKE [20,51,59]. Indeed, these attacks uncovered a powerful tool: using higher-dimensional representations, we can efficiently represent a one-dimensional isogeny of non-smooth degree by embedding it in a higher-dimensional isogeny of smooth degree between products of elliptic curves. The efficiency of the protocol depends on the embedding dimension of the isogenies used in this representation.

The first variant of SQIsign that used the SIDH attacks constructively was SQIsignHD [24], which relied on four-dimensional representations to obtain an impressive speed-up in the signing time and even smaller signatures than SQIsign. Verification, however, suffered from the heavy cost of computing four-dimensional isogenies. A wave of new protocols — SQIsign2D-West [9], SQIsign2D-East [54] and SQIPrime [34] — presented new variants of SQIsign making use of two-dimensional representations. These two-dimensional variants of SQIsign achieve signature sizes comparable to SQIsignHD, faster signing than SQIsign (though slower than SQIsignHD), and verification faster than both. A refinement of SQIsign2D-West is currently the official version of SQIsign submitted to the Round 2 of NIST standardization [1]; from now on, when referring to just SQIsign, we will mean this version.

Another drawback that affects all variants of SQIsign is their design complexity. The intricate signature process complicates the description of the scheme, and this reflects both in the security analysis and in the flexibility of the design. The former issue leads to security proofs requiring specific ad-hoc oracles and assumptions. The latter makes it hard to use these schemes as a building block for more elaborated protocols. Despite its extremely compact signatures, after

several years since its initial publication, only a few advanced functionalities based on SQIsign have been proposed; see, for example, [16,58].

High-dimensional isogenies were used in a different manner by Leroux in [50] to introduce an efficient Verifiable Unpredictable Function (VUF), from which a Verifiable Random Function and the first isogeny-based hash-and-sign signature scheme can be derived. The function underlying Leroux’s construction computes the codomain of an isogeny of given kernel, where the kernel has a fixed large prime order. The security thus relies on the hardness of computing such a function without the knowledge of the endomorphism ring of the domain curve.

Contributions. We present a new isogeny-based identification protocol, which we transform into a signature scheme PRISM-sig via the hash-and-sign paradigm. The security of both schemes relies on the fact that computing a large prime degree isogeny from an elliptic curve E is conjectured to be hard *without knowledge of the endomorphism ring of E* . The identification protocol is very simple:

1. The prover samples a secret key $\phi_{sk} : E_0 \rightarrow E_{vk}$ with corresponding public key E_{vk} .
2. A verifier challenges the prover with a large prime q .
3. The prover replies with a two-dimensional representation of a degree- q isogeny from E_{vk} .

The simplicity of the above protocol and the derived signature scheme, particularly when compared to SQIsign and its variants, improves the flexibility of the scheme itself and makes it easier to assess its security. In particular, we obtain an isogeny-based signature scheme that is secure in the standard model. In view of the simplicity, we also expect our construction to be a useful building block in other, more advanced schemes.

Our contributions are summarized as follows:

- In Section 3, we construct a new efficient identification protocol, which we call PRISM-id. Unlike other isogeny-based ID protocols, PRISM-id is *not* a Σ -protocol: this leads to smaller communication costs and more efficient computations. After constructing an appropriate hash function, we obtain a signature scheme (PRISM-sig) based on our identification protocol via the hash-and-sign paradigm. The resulting signature scheme is simple, flexible, compact, and efficient.
- In Section 4, we prove the security of both the identification scheme and the signature scheme in the standard model, showing that their hardness is linked to well-understood problems in isogeny-based cryptography. For the signature scheme, we also prove security under a weaker assumption in the random oracle model.
- In Section 5, we describe a variant of PRISM-sig, which we argue to be strongly unforgeable under some mildly-stronger assumptions. Notably, our variant achieves strong unforgeability while requiring only two-dimensional signature isogenies.

- In [Section 6](#), we present a modified version of protocol PRISM-sig in which a random salt is added to the message before hashing. We call this variant salt-PRISM. The use of a salt allows us to decrease the value of a needed for our signatures, resulting in better efficiency without compromising security.
- In [Section 7](#), we propose concrete parameters for our schemes for NIST Level I, III, and V security, and we provide an optimized implementation in C of salt-PRISM. We compare its efficiency and key/signature sizes with other variants of SQIsign. We also compare PRISM-id, PRISM-sig, and salt-PRISM, and observe that PRISM-id has a faster verification and much lower communication costs. The reason is that to instantiate a secure signature scheme via the hash-and-sign paradigm we need to avoid hash collisions, forcing us to select larger parameters for the signatures compared to PRISM-id.

Related work. Our construction shares strong conceptual similarities with Leroux’s verifiable unique function (VUF) [50] as they both rely on the hardness of computing large degree isogenies without the knowledge of the endomorphism ring. The main difference lies in the degree of the response isogenies: in [50], the isogenies have fixed degree, whereas in our case *the degree is the challenge*, and it thus changes across executions. As a result, our choice of parameters is less constrained, and our security is arguably better. Indeed, the response isogeny has kernel defined over a field extension of exponential degree with overwhelming probability (whereas in [50] the kernel is defined over a small field extension), which provides us with additional security guarantees even if there were breakthroughs in the computation of large prime degree isogenies.

Compared to SQIsign and its variants, our scheme is significantly simpler. Not only does this simplicity make it easier to analyze and implement the protocol, but it also leads to faster signing times. The signing procedure of salt-PRISM requires computing fewer two-dimensional isogenies compared to all the SQIsign variants, and no one-dimensional isogenies at all, leading to a $1.4\times$ to $1.6\times$ speedup when compared to the sound 2 NIST submission of SQIsign. Thanks to the addition of the salt, salt-PRISM allows for smaller parameters with respect to PRISM-sig, and this has a positive impact on verification: when compared with SQIsign, the verification time goes from $1.2\times$ slower to $1.01\times$ faster at level V. The signature size is very compact, and is comparable with most other isogeny-based protocols.

Acknowledgements. We thank Sina Schaeffler for her helpful comments and remarks. We thank Filipe Casal for his valuable comments on the implementation and security of the hash function. We thank Tibor Jager and Greg Zaverucha for their valuable comments on the salted variant. This work was supported in part by the European Research Council (ERC) under the European Union’s Horizon 2020 research and innovation programme (grant agreement ISOCRYPT - No. 101020788), by the Research Council KU Leuven grant C14/24/099, by CyberSecurity Research Flanders with reference number VR20192203, by Fonds voor Wetenschappelijk Onderzoek (FWO) with reference number 1138925N, and

by Engineering and Physical Sciences Research Council (EPSRC) grant number EP/V011324/1. This work was supported in part by SNSF Consolidator Grant CryptonIs 213766, as well as the European Research Council under grant No. 101116169 (AGATHA CRYPTY).

2 Preliminaries

In this section, we recall some background knowledge about the Deuring correspondence, computation of isogenies in dimension two and different variants of the SQIsign protocol. We assume some familiarity with elliptic curves and their isogenies, and refer the reader to [26, 61] for more information. From this point onwards p is a prime with $p \equiv 3 \pmod{4}$ and Ell_p denotes the set of supersingular elliptic curves defined over $\mathbb{F}_{p^2}$ up to isomorphism.

2.1 Quaternion algebras and the Deuring correspondence

We start by introducing quaternion algebras. A *quaternion algebra* over $\mathbb{Q}$ is a division algebra defined by $\mathbb{Q} + \mathbb{Q}i + \mathbb{Q}j + \mathbb{Q}k$, where $i^2 = a$, $j^2 = b$, $ij = -ji = k$ for some $a, b \in \mathbb{Q}^*$. We denote it by $H(a, b)$. We say that $H(a, b)$ is *ramified* at a place v of $\mathbb{Q}$ if the extension of scalars $H(a, b) \otimes_{\mathbb{Q}} \mathbb{Q}_v$ is not isomorphic to the algebra of 2×2 matrices over $\mathbb{Q}_v$. Up to isomorphism, for a given prime p there exists a unique quaternion algebra ramified exactly at p and ∞ , which we denote by $\mathcal{B}_{p, \infty}$. Since we assume $p \equiv 3 \pmod{4}$, we can choose a basis such that $i^2 = -1$ and $j^2 = -p$.

For a given element $\alpha = a + bi + cj + dk \in \mathcal{B}_{p, \infty}$ we define its conjugate $\bar{\alpha} := a - bi - cj - dk$ and its *reduced norm* $\text{nrd}(\alpha) := \alpha \bar{\alpha}$. A *fractional ideal* in $\mathcal{B}_{p, \infty}$ is a $\mathbb{Z}$ -submodule of rank 4. An *order* $\mathcal{O}$ is a fractional ideal that is also a subring. An order is *maximal* if it is not properly contained in any other order. Let I be a fractional ideal in $\mathcal{B}_{p, \infty}$. We define the left order of I to be $\mathcal{O}_L(I) := \{\alpha \in \mathcal{B}_{p, \infty} \mid \alpha I \subset I\}$. We can similarly define the right order $\mathcal{O}_R(I)$ of a fractional ideal I , and I is called a *connecting ideal* for $\mathcal{O}_L(I)$ and $\mathcal{O}_R(I)$. If I is contained in its left order (or, equivalently, in its right order) then it is an *integral ideal*, or just an *ideal* for short.

For a fractional ideal I , we denote its conjugate by $\bar{I} = \{\bar{\alpha} \mid \alpha \in I\}$. The reduced norm of an ideal I , denoted by $\text{nrd}(I)$, is defined as the gcd of the reduced norms of the elements of I . For a maximal order $\mathcal{O}$, any left $\mathcal{O}$ -ideal I can be written as $I = \mathcal{O}\alpha + \mathcal{O}\text{nrd}(I)$ for some $\alpha \in I$. Two ideals I and J are *equivalent* if there exists $\beta \in \mathcal{B}_{p, \infty}^*$ such that $I = J\beta$. We denote equivalence by $I \sim J$. A more detailed discussion of quaternion algebras can be found in [66].

The Deuring Correspondence. Deuring [31] showed a categorical equivalence between maximal orders in $\mathcal{B}_{p, \infty}$ and supersingular elliptic curves defined over $\mathbb{F}_{p^2}$. This equivalence is known as the *Deuring correspondence*. Under this correspondence, to each maximal order $\mathcal{O}$ of $\mathcal{B}_{p, \infty}$ we can associate a supersingular elliptic curve E over $\mathbb{F}_{p^2}$ such that $\text{End}(E) \cong \mathcal{O}$. An isogeny $\varphi : E_1 \rightarrow E_2$

corresponds to an ideal I_φ , where $\mathcal{O}_L(I_\varphi) \cong \text{End}(E_1)$ and $\mathcal{O}_R(I_\varphi) \cong \text{End}(E_2)$. Moreover, $\deg(\varphi) = \text{nrd}(I_\varphi)$.

Example 1. Since $p \equiv 3 \pmod{4}$, the elliptic curve $E_0 : y^2 = x^3 + x$ defined over $\mathbb{F}_{p^2}$ is supersingular. We can define endomorphisms $\iota : (x, y) \mapsto (-x, \sqrt{-1}y)$ and $\pi : (x, y) \mapsto (x^p, y^p)$ of E_0 , where $\sqrt{-1}$ is a fixed square root of -1 in $\mathbb{F}_{p^2}$. We have the following isomorphism of rings:

$$\begin{aligned} \mathcal{O}_0 &:= \mathbb{Z} \left\langle 1, i, \frac{i+j}{2}, \frac{1+k}{2} \right\rangle \longrightarrow \text{End}(E), \\ a + bi + cj + dk &\longmapsto a + b\iota + c\pi + d\iota\pi. \end{aligned}$$

Throughout the paper, we will always denote by E_0 the curve $y^2 = x^3 + x$.

Pushforward Isogenies and Ideals. Consider an isogeny $\varphi_1 : E \rightarrow E_1$ and a separable isogeny $\varphi_2 : E \rightarrow E_2$ of degree coprime to $\deg(\varphi_1)$. We denote by $[\varphi_1]_*\varphi_2 : E_1 \rightarrow E'$ the pushforward isogeny of φ_2 under φ_1 , i.e., the separable isogeny such that $\ker([\varphi_1]_*\varphi_2) = \varphi_1(\ker(\varphi_2))$; see [Figure 1](#).

$$\begin{array}{ccc} E & \xrightarrow{\varphi_1} & E_1 \\ \varphi_2 \downarrow & & \downarrow [\varphi_1]_*\varphi_2 \\ E_2 & & E' \end{array}$$

Fig. 1. Pushforward isogeny of φ_2 under φ_1 .

Under the Deuring correspondence, we can define the pushforward of I_{φ_2} under I_{φ_1} as the left ideal of $\mathcal{O}_R(I_{\varphi_1})$ corresponding to the isogeny $[\varphi_1]_*\varphi_2$, and we denote it by $[I_{\varphi_1}]_*I_{\varphi_2}$. By [\[27, Lemma 3\]](#) it can be computed as

$$[I_{\varphi_1}]_*I_{\varphi_2} = \frac{1}{\text{nrd}(I_{\varphi_1})} \overline{I_{\varphi_1}}(I_{\varphi_1} \cap I_{\varphi_2}). \quad (1)$$

We summarize the Deuring correspondence in [Table 1](#).

2.2 Kani's Lemma

Kani's Lemma [\[45\]](#) gives a criterion to compute isogenies of dimension one using isogenies of dimension two. It was at the heart of the recent SIDH attacks [\[20, 51, 59\]](#), but it quickly turned into a powerful building block for isogeny-based protocols. We will extensively use it in this work. Our formulation follows [\[51\]](#).

Theorem 1 (Kani). *Let d_1, d_2 and N be pairwise coprime integers such that $N = d_1 + d_2$, and let E_0, E_1, E_2 , and E_3 be elliptic curves connected by the following diagram of isogenies:*

Supersingular elliptic curves	Quaternions
Supersingular j -invariants $j(E) \in \mathbb{F}_{p^2}$ (up to Galois conjugacy)	Maximal orders $\mathcal{O} \cong \text{End}(E)$ in $B_{p,\infty}$ (up to isomorphism)
(E_1, φ) with $\varphi : E \rightarrow E_1$	I_φ integral left $\mathcal{O}$ -ideal and right $\mathcal{O}_1$ -ideal
$\theta \in \text{End}(E)$	Principal ideal $\mathcal{O}\theta$
$\deg(\varphi)$	$\text{nrd}(I_\varphi)$
$\widehat{\varphi}$	$\overline{I_\varphi}$
$\varphi : E \rightarrow E_1, \psi : E \rightarrow E_1$	Equivalent ideals $I_\varphi \sim I_\psi$
$\tau \circ \rho : E \rightarrow E_1 \rightarrow E_2$	$I_{\tau \circ \rho} = I_\rho \cdot I_\tau$

Table 1. The Deuring correspondence, a summary given in [28].

$$\begin{array}{ccc}
E_0 & \xrightarrow{\varphi_1} & E_1 \\
\downarrow \psi_2 & & \downarrow \varphi_2 \\
E_2 & \xrightarrow{\psi_1} & E_3
\end{array}$$

such that $\deg(\varphi_1) = \deg(\psi_1) = d_1$, $\deg(\varphi_2) = \deg(\psi_2) = d_2$ and $\varphi_2 \circ \varphi_1 = \psi_1 \circ \psi_2$. Then the map

$$\Phi = \begin{pmatrix} \varphi_1 & \widehat{\varphi_2} \\ -\psi_2 & \psi_1 \end{pmatrix} : E_0 \times E_3 \rightarrow E_1 \times E_2$$

is an isogeny of (principally polarized) abelian varieties with kernel

$$\ker(\Phi) = \{(\widehat{\varphi_1}(P), \varphi_2(P)) \mid P \in E_1[N]\} \cong \frac{\mathbb{Z}}{N\mathbb{Z}} \times \frac{\mathbb{Z}}{N\mathbb{Z}}.$$

Assuming that N is powersmooth, or N is smooth and all N -torsion points are rational, the isogeny Φ can be efficiently evaluated at any point on $E_0 \times E_3$. For example, if $N = 2^a$ for some $a \geq 1$ then one can use the algorithms given in [25]. In this case, the generators of the kernel defining Φ encode an efficient two-dimensional representation of φ_1 .

2.3 Ideal To Isogeny Translation

Translating ideals into the corresponding isogenies under the Deuring correspondence is a fundamental task in isogeny-based cryptography. A first breakthrough was made in 2014 by Kohel, Lauter, Petit and Tignol [47], who introduced the KLPT algorithm that was at the heart of the first version of the SQIsign protocol. The KLPT algorithm can be turned into a polynomial-time method for converting ideals to isogenies, but this is usually inefficient in practice, due to the large degree of the auxiliary isogeny appearing in the process, whose kernel elements are

in general defined over large field extensions only. Recently, higher dimensional isogenies gave a new algorithmic tool for converting ideals to isogenies. In particular, the Round 2 version of SQIsign is based on the `IdealTolsogeny` algorithm introduced in SQIsign2D-West [9]. This algorithm can efficiently translate *any* left $\mathcal{O}_0$ -ideal, where $\mathcal{O}_0 = \text{End}(E_0)$, to the corresponding isogeny originating at E_0 when working over $\mathbb{F}_{p^2}$ with $p = f2^e - 1$, for some small odd $f > 0$. This algorithm was later improved by the development of the `Qlapoti` algorithm [17], used as a subroutine in `IdealTolsogeny`. In this work, we use this variant of the `IdealTolsogeny` algorithm, which we sketch below.

Let I be a left $\mathcal{O}_0$ -ideal, which we assume being of smallest norm within its equivalence class, and let $n = \text{nrd}(I)$. The goal is to find two ideals I_1, I_2 equivalent to I , such that

$$\text{nrd}(I_1) + \text{nrd}(I_2) = 2^e$$

where e is (smaller than) the 2-valuation of p . Since I_1, I_2 must be equivalent to I , there will exist $\beta_k \in I$ for $k = 1, 2$ such that $I_k = I\beta_k/n$. The above equation can thus be written in terms of the β_i as

$$\text{nrd}(\beta_1) + \text{nrd}(\beta_2) = 2^e \cdot n. \quad (2)$$

By writing $I = \langle n, \alpha \rangle$ for some α of relatively small norm, β_i is written explicitly as $\gamma_i \cdot n + \alpha$, where $\gamma_i \in \mathbb{Z}[i]$. The norm equation is then spelled out completely, and solved in two steps. In the first step, a solution is found mod n ; a lattice reduction ensures that the solution is small. In the second step, this small solution is put back in the original equation, which can now be expressed as a sum-of-squares problem and solved using Cornacchia's algorithm.

Once the two elements β_1, β_2 (and in turn the ideals I_1, I_2) are found, the ideal I is recovered using 2-dimensional isogenies in a very similar way to the original `IdealTolsogeny` algorithm.

2.4 Identification Protocols and Digital Signatures

Identification protocols and digital signatures are basic cryptographic building blocks that share some similarities, for example a digital signature scheme implies an identification protocol (see [63, Section 10.3]), but have important differences in their definition and security notions.

Definition 1 (Definition 18.1 [15]). *An identification protocol is a triple of probabilistic polynomial-time algorithms $(\text{KeyGen}, \text{Pvr}, \text{Vrf})$ such that:*

- $(\text{vk}, \text{sk}) \leftarrow \text{KeyGen}(1^\lambda)$, is a probabilistic key generation algorithm, that takes as input a security parameter λ , and outputs a pair (vk, sk) , where vk is called the verification key and sk is called the secret key;
- $\text{Pvr}(\text{sk})$, is an interactive protocol algorithm, called the prover, that takes as input the secret key sk ;

- $\text{accept/reject} \leftarrow \text{Vrf}(\text{vk})$ is a probabilistic interactive protocol algorithm, called the verifier, that takes as input the verification key vk and outputs accept or reject .

We model the interaction with the following notation: $\text{output} \leftarrow \text{Vrf}(\text{vk}) \rightleftharpoons \text{Pvr}(\text{sk})$.

We say that the identification scheme is *correct* if Vrf outputs accept with probability 1 over the choice of $(\text{vk}, \text{sk}) \leftarrow \text{KeyGen}(1^\lambda)$ and over the randomness in the involved algorithms, i.e., if

$$\Pr[\text{accept} \leftarrow \text{Vrf}(\text{vk}) \rightleftharpoons \text{Pvr}(\text{sk})] = 1.$$

We consider security against active attacks following Boneh and Shoup [15, Definition 18.8].

Definition 2 ([15]). Consider the following three phase impersonator game for an adversary $\mathcal{A}$:

- **Setup:** a key pair $(\text{vk}, \text{sk}) \leftarrow \text{KeyGen}(1^\lambda)$ is generated and the adversary $\mathcal{A}$ receives vk ;
- **Probing:** the adversary $\mathcal{A}$ can interact multiple times with an honest prover $\text{Pvr}(\text{sk})$ and store the interaction outputs in a state st :

$$\text{st} \leftarrow \mathcal{A}(\text{vk}, \text{st}) \rightleftharpoons \text{Pvr}(\text{sk});$$

- **Impersonation:** the adversary $\mathcal{A}(\text{vk}, \text{st})$ interacts with an honest verifier $\text{Vrf}(\text{vk})$:

$$\text{output}_{\mathcal{A}} \leftarrow \text{Vrf}(\text{vk}) \rightleftharpoons \mathcal{A}(\text{vk}, \text{st}).$$

The adversary wins if the verifier accepts, i.e., $\text{output}_{\mathcal{A}} = \text{accept}$.

An identification protocol is secure against active attacks if for any PPT adversary $\mathcal{A}$ playing the impersonator game it is not able to authenticate with non-negligible probability, i.e., $\Pr[\text{output}_{\mathcal{A}} = \text{accept}] = \text{negl}(\lambda)$.

A particular class of identification protocols are Σ -protocols. These are three-round interactive protocols usually referred to as the commitment, challenge and response phase, represented by a transcript $(\text{com}, \text{chall}, \text{resp})$. An example of a Σ -protocol is the identification protocol underlying SQIsign, that we recall in Section 2.5. Due to the Fiat–Shamir transform [37], secure Σ -protocols can be efficiently transformed into secure signature schemes. Thus, many signature schemes, such as SQIsign, are constructed in this manner.

Definition 3. A digital signature scheme consists of three probabilistic polynomial-time algorithms $(\text{KeyGen}, \text{Sign}, \text{Vrf})$ such that:

- $(\text{vk}, \text{sk}) \leftarrow \text{KeyGen}(1^\lambda)$: on input a security parameter λ , the key generation algorithm outputs a pair of verification and signing keys (vk, sk) ;
- $\sigma \leftarrow \text{Sign}(\text{sk}, \text{msg})$: on input a signing key sk and a message msg , the signing algorithm outputs a signature σ ;

- $\text{accept/reject} \leftarrow \text{Vrf}(\text{vk}, \text{msg}, \sigma)$: on input a verification key vk , a message msg and a signature σ , the verification algorithm outputs *accept* or *reject*.

A signature scheme is correct if, given $(\text{vk}, \text{sk}) \leftarrow \text{KeyGen}(1^\lambda)$, for any message msg and signature $\sigma \leftarrow \text{Sign}(\text{sk}, \text{msg})$ a run of $\text{Verify}(\text{vk}, \text{msg}, \sigma)$ outputs *accept* with probability 1.

Definition 4. Let $\text{ATK} \in \{\text{EUF}, \text{SUF}\}$. A digital signature is secure in the ATK-CMA model if for any PPT adversary $\mathcal{A}$ playing the game from Figure 2, it holds

$$\text{Adv}_{\mathcal{A}}^{\text{ATK-CMA}}(\lambda) = \Pr [G^{\text{ATK}}(\mathcal{A}) = 1] = \text{negl}(\lambda). \quad (3)$$

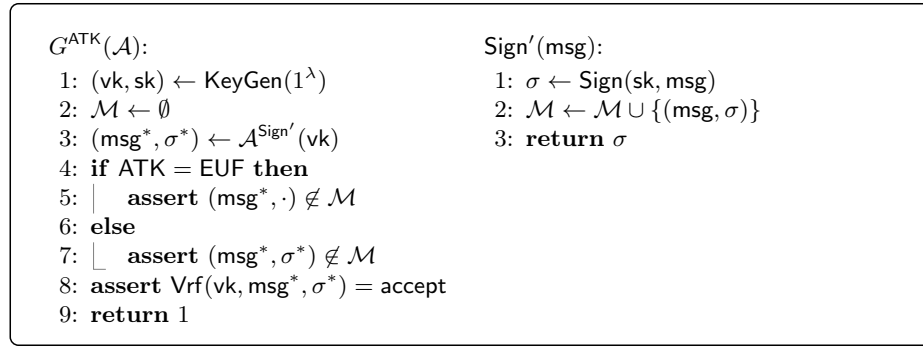


Fig. 2. Security game for the EUF-CMA (SUF-CMA) property

2.5 SQIsign and its variants

SQIsign [27] is a signature scheme derived from an isogeny-based Σ -protocol. It is represented by the following diagram:

$$\begin{array}{ccc}
 E_0 & \xrightarrow{\psi_{\text{com}}} & E_{\text{com}} \\
 \downarrow \phi_{\text{sk}} & & \searrow \phi_{\text{chall}} \\
 E_{\text{vk}} & \xrightarrow{\sigma_{\text{resp}}} & E_{\text{chall}}
 \end{array} \quad (4)$$

Here, ϕ_{sk} is the secret key with corresponding verification key E_{vk} , E_{com} is the commitment, and ϕ_{chall} is the challenge. As the prover knows the secret isogeny ϕ_{sk} , they have knowledge of the endomorphism ring of E_{vk} . In this way, the prover is the only party capable of producing a valid response σ_{resp} connecting E_{vk} to the challenge curve E_{chall} . To construct the response isogeny, the prover finds the ideal I corresponding to the composition $\phi_{\text{chall}} \circ \psi_{\text{com}} \circ \phi_{\text{sk}}$, then it finds

an *equivalent* ideal J not factoring through the secret key ϕ_{sk} , and translates J to its corresponding isogeny σ_{resp} . In the original SQIsign protocol [27] this is done with (a variant of) the KLPT algorithm, that finds J of smooth norm. Instead, in the subsequent *higher-dimensional variants* [9, 34, 24, 54], they can pose milder restrictions on $\text{nrd}(J)$ and use Kani’s Lemma (Section 2.2) to represent the isogeny.

High Degree Oracles and HVZK property. To show that a Σ -protocol satisfies the Honest-Verifier Zero Knowledge property we need to show how to simulate a valid transcript $(\text{com}, \text{chall}, \text{resp})$ in polynomial time, without access to the secret key. The natural simulation strategy for the Σ -protocol underlying SQIsign in (4) is to first generate at random $\sigma_{\text{resp}} : E_{\text{vk}} \rightarrow E_{\text{chall}}$, then $\hat{\phi}_{\text{chall}} : E_{\text{chall}} \rightarrow E'$,¹⁰ and finally define $E_{\text{com}} := E'$.

In the original SQIsign identification protocol both the challenge isogeny ϕ_{chall} and the response isogeny σ_{resp} have smooth degree, thus they can efficiently be sampled to simulate a transcript, though arguing the indistinguishability of the transcript requires an *ad-hoc* assumption. This is not true for the high-dimensional variants: the response isogeny σ_{resp} has a larger non-smooth degree that can be efficiently represented via high-dimensional isogenies to allow the verifier to compute it. However, this representation cannot be provided without the knowledge of the endomorphism ring of the public curve E_{vk} . To overcome this problem, all the security proofs of the higher-dimensional variants rely on auxiliary oracles that provide efficient representations of uniformly random non-smooth degree isogenies. These oracles can be characterized as providing:

- isogenies of a fixed degree given as input, FIDIO [9, Definition 23] and AIO [34, Definition 4];
- isogenies of a random degree satisfying specific conditions, like RUGDIO [24, Definition 20] and RUNDIO [54, Definition 2];
- uniformly random isogenies of uniformly random bounded degree, like RADIO [24, Definition 41] and UTO [9, Definition 21];
- efficient representation of isogenies given their non-smooth kernel, like FIXDIO [50, Definition 4], RUCODIO [34, Definition 3] and RUCGDIO [34, Definition 2].

Though these oracles vary in flavor, they are all needed for the same core reason: to the best of our knowledge, there are no efficient algorithms to compute large prime degree isogenies without leveraging the knowledge of the endomorphism ring of the domain curve.

Although, we do not know how to construct any of these oracles in polynomial time, it is conjectured that a bounded number of queries do not provide any help in compromising the security of the schemes, e.g., by recovering non-trivial endomorphisms. This was first argued in [24, Section 5.3].

¹⁰ The order of computing ϕ_{chall} and σ_{resp} may be inverted in some variants, but the simulation strategies rely on the same machinery.

3 Identification Protocol and Digital Signature Scheme

In this section, we present a new identification scheme built from the conjectured hardness of constructing large prime degree isogenies from E , without knowledge of the endomorphism ring $\text{End}(E)$. After constructing an appropriate hash function, this identification protocol can easily be transformed into a simple signature scheme, built from the same subroutines. We can view this last scheme as a *hash-and-sign* signature where the trapdoor one-way function consists of computing a large prime degree isogeny with domain E_{vk} . The trapdoor one-way function is the degree map applied to isogenies of large prime degree with domain E_{vk} . Indeed, given an isogeny, its degree can be efficiently recovered by anyone; however, inverting the trapdoor consists of sampling a large prime degree isogeny, a task considered to be hard without access to the endomorphism ring of E_{vk} .

3.1 Identification Protocol

The core idea behind our identification protocol is that computing isogenies of large prime degree from a random elliptic curve with unknown endomorphism ring is believed to be hard. Indeed, the best known algorithm for computing an isogeny of prime degree q runs in $\tilde{O}(\min(q^2, p^{1/2}))$ (see [Section 4.4](#)). However, this task significantly simplifies with knowledge of the endomorphism ring of the domain elliptic curve. From these two observations, we can now construct an identification scheme: a party can show their knowledge of the endomorphism ring of a given curve E_{vk} by publishing an isogeny of large prime degree $\varphi : E \rightarrow E'$. To instantiate the protocol we fix a base prime $p \equiv 3 \pmod{4}$ and an integer a such that we have $\mathbb{F}_{p^2}$ -rational 2^a -torsion on any supersingular elliptic curve on $\mathbb{F}_{p^2}$ (when working with models having $(p+1)^2$ rational points). For the same number a , we define Primes_a to be the set of primes of exactly a bits, i.e., primes q such that $2^{a-1} < q < 2^a$.

The identification scheme, referred to as PRISM-id and depicted in [Figure 3](#), relies on the following subroutines:

- $\varphi \leftarrow \text{GenIsogeny}(E, \phi, q)$: on input a supersingular elliptic curve E , an isogeny $\phi : E_0 \rightarrow E$ (that gives access to the endomorphism ring of E) and a prime q , returns an efficient representation of a cyclic isogeny $\varphi : E \rightarrow E'$ of degree $q(2^a - q)$; furthermore, φ is uniformly distributed among all cyclic isogenies of degree $q(2^a - q)$ from E .
- $\text{accept/reject} \leftarrow \text{VerIsogeny}(\varphi, E, q)$: on input an efficient representation of an isogeny $\varphi : E \rightarrow E'$, checks whether it has degree $q(2^a - q)$.

Security. The security of these protocols rests on the following assumption: given many isogenies $\varphi_i : E \rightarrow E_i$ of degree $q_i(2^a - q_i)$ with q_i an a -bit prime number, the verifier does not learn any information about the endomorphism ring of E . This assumption is plausible as there are already efficient ways to compute large

Prover	Verifier
$\text{sk} = \phi_{\text{sk}}$	$\text{vk} = E_{\text{vk}}$
	$\xleftarrow{q} \text{ Sample } q \in \text{Primes}_a$
$\varphi \leftarrow \text{GenIsogeny}(\text{vk}, \text{sk}, q)$	
	$\xrightarrow{\varphi}$
	if $\text{VerIsogeny}(\varphi, E_{\text{vk}}, q)$
	return accept

Fig. 3. PRISM-id identification scheme.

(yet smooth) degree isogenies from a given curve, regardless of any knowledge of the endomorphism ring. Moreover, because Primes_a is defined as the set of primes having a bits exactly, the cofactor $2^a - q$ cannot be another a -bit prime, so seeing isogenies of degrees corresponding to $q_1, \dots, q_r \in \text{Primes}_a$ does not help to respond to a degree q_{r+1} that has not been queried before. The security of the schemes and the underlying assumptions are discussed in greater detail in [Section 4](#).

Remark 1. The degree of the isogeny ϕ has the form $q(2^a - q)$ so that ϕ can be represented in two dimensions (rather than four) by using Kani’s lemma, which results in a more efficient protocol. The verification procedure thus involves computing a two-dimensional isogeny, which is significantly more efficient than the analogous computation in dimension four.

For a slightly different perspective, write the response isogeny ϕ as the composition $\phi = \phi_{2^a - q} \circ \phi_q$, where $\deg \phi_{2^a - q} = 2^a - q$ and $\deg \phi_q = q$. The isogeny ϕ_q can be interpreted as the “real” response isogeny, whereas $\phi_{2^a - q}$ is just an auxiliary isogeny that is needed to obtain a two-dimensional representation.

It is also possible to instantiate PRISM with higher-dimensional isogenies, which would eliminate the need for an auxiliary isogeny and would lead to an even more compact protocol (four-dimensional representations require smaller-order torsion points), but would also come at the cost of a much slower verification procedure (extrapolating from the software given in [62], four-dimensional isogenies in SQIsign-HD verification are roughly 7 times slower than our verification for similar parameter sets).

3.2 Signature scheme

If we also have access to a collision resistant hash function on the set of large primes $H_{\text{prime}} : \{0, 1\}^* \rightarrow \text{Primes}_a$ we can define a digital signature, called PRISM-sig, using the same building blocks of the identification protocol based on the well-known hash-and-sign paradigm in which we hash a given message to a set of suitable prime degrees, and give an isogeny of that degree as its signature. This is explained in [Figure 4](#). We highlight the simplicity of this scheme: it directly reduces to conjectured hard problems in isogeny computation, simplifying its analysis in terms of both efficiency and security.

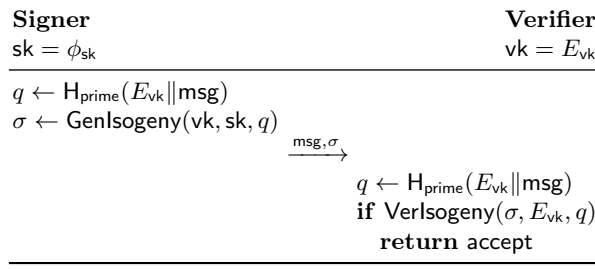


Fig. 4. PRISM-sig signature scheme.

In [Section 3.3](#) we carefully define the necessary subroutines for the two schemes.

3.3 Subroutines based on IdealTolsogeny

In the previous sections, we introduced our constructions for an identification protocol and signature scheme. However, we did not specify how key generation or the subroutines `GenIsogeny` and `VerIsogeny` are constructed. For the signature scheme, we also need to specify the hash function H_{prime} . We present here our main construction, which relies on the `IdealTolsogeny` algorithm introduced in [Section 2.3](#). We note however that alternative choices are possible thanks to the high modularity of our schemes (see e.g. the appendix of the original paper [8]).

In what follows, we fix $p = f2^e - 1$ for some small integer $f > 0$. This choice is motivated by the need to access rational 2^e -torsion during the `IdealTolsogeny` algorithm. Let $E_0 : y^2 = x^3 + x$ be the supersingular curve with j -invariant 1728, and fix a basis P_0, Q_0 of $E_0(\mathbb{F}_{p^2})[2^a]$, for some $a \leq e$.

Key generation. The key generation procedure is performed by the algorithm `KeyGen` [1, Algorithm 4.1], with the only difference that `IdealTolsogeny` is replaced by `Qlapoti` as the main subroutine. On input the security parameter λ , `KeyGen` outputs the public and secret key pair (vk, sk) as follows. First, sample a random ideal I_{sk} of random prime norm N_{sk} , such that the distribution of ϕ_{sk} is statistically close to uniform. Use `IdealTolsogeny` to compute the isogeny $\phi_{sk} : E_0 \rightarrow E_{vk}$ corresponding to the ideal I_{sk} . Then deterministically compute generators P_{vk}, Q_{vk} of $E_{vk}(\mathbb{F}_{p^2})[2^a]$, and the matrix M_{sk} sending $(\phi_{sk}(P_0), \phi_{sk}(Q_0))$ into (P_{vk}, Q_{vk}) , i.e., the matrix mod 2^a given by

$$M_{sk} = \begin{pmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{pmatrix}$$

such that $P_{vk} = [m_{11}]\phi_{sk}(P_0) + [m_{12}]\phi_{sk}(Q_0)$ and $Q_{vk} = [m_{21}]\phi_{sk}(P_0) + [m_{22}]\phi_{sk}(Q_0)$. The public key is set to be E_{vk} , and the secret key is (E_{vk}, I_{sk}, M_{sk}) .

Remark 2. Concretely, the deterministic basis (P_{vk}, Q_{vk}) is computed with the algorithm `TorsionBasisToHint` [1, Algorithm 2.1], together with a hint hint_{vk} . The

hint is then provided together with the public key to speed up the deterministic basis generation on the verifier side, as done in [1]. We note however that this choice is done purely to increase efficiency. Indeed, anyone can generate the correct basis (P_{vk}, Q_{vk}) without hints. For the sake of simplicity, the hints are, from this point onwards, not explicitly included in our algorithms.

Isogeny Generation. The algorithm $\text{GenIsogeny}(vk, sk, q)$, given in [Algorithm 1](#), is constructed as follows. Parse vk as E_{vk} and sk as (E_{vk}, I_{sk}, M_{sk}) . Compute the deterministic basis (P_{vk}, Q_{vk}) . As the endomorphism ring of E_0 is known (see [Example 1](#)), via the secret isogeny $\phi_{sk} : E_0 \rightarrow E_{vk}$ we have knowledge of the endomorphism ring of E_{vk} . We can use this to generate an isogeny $\sigma : E_{vk} \rightarrow E_{sig}$ of degree $q(2^a - q)$. More precisely, we compute a two-dimensional representation of the isogeny $\sigma : E_{vk} \rightarrow E_{sig}$, consisting of the codomain curve E_{sig} and the image of the basis (P_{vk}, Q_{vk}) of $E_{vk}[2^a]$ through σ .

The first step is to generate an $\mathcal{O}_0$ -ideal I_{chall} of norm $q(2^a - q)$. This is done using the `RandomIdealGivenNorm` algorithm [1, Algorithm 3.10]. Since (with overwhelming probability) I_{sk} and I_{chall} have coprime norm, we can compute the pushforward of I_{chall} through the secret ideal I_{sk} using [Equation \(1\)](#):

$$I_\sigma = [I_{sk}]_* I_{chall} = \frac{1}{N_{sk}} \overline{I_{sk}}(I_{sk} \cap I_{chall}).$$

From this, we see that the ideal corresponding to the isogeny $\varphi = \sigma \circ \phi_{sk} : E_0 \rightarrow E_{sig}$ is given by $I = I_{sk} I_\sigma = I_{sk} \cap I_{chall}$. Using `IdealToIsogeny` we can obtain a representation of φ and compute the images $\varphi(P_0), \varphi(Q_0)$. Finally, we obtain a two-dimensional representation of σ as follows. For $P \in E_{vk}[2^a]$ we have

$$\sigma(P) = \frac{1}{N_{sk}} (\varphi \circ \widehat{\phi_{sk}})(P).$$

From the definition of M_{sk} we see that

$$\frac{1}{N_{sk}} \widehat{\phi_{sk}}(P_{vk}) = [m_{11}]P_0 + [m_{12}]Q_0,$$

and hence $\sigma(P) = [m_{11}]\varphi(P_0) + [m_{12}]\varphi(Q_0)$; similarly, $\sigma(Q) = [m_{21}]\varphi(P_0) + [m_{22}]\varphi(Q_0)$. The kernel of the two-dimensional isogeny embedding σ is now given by

$$\langle ([q]P_{vk}, \sigma(P)), ([q]Q_{vk}, \sigma(Q)) \rangle = \langle (P_{vk}, [q^{-1}]\sigma(P)), (Q_{vk}, [q^{-1}]\sigma(Q)) \rangle.$$

The right hand side can be obtained with the same amount of point multiplications (it is enough to multiply the m_{ij} by q^{-1}) and makes verification easier; we thus return $(E_{sig}, P_{sig} = [q^{-1}]\sigma(P), Q_{sig} = [q^{-1}]\sigma(Q))$.

Remark 3. The `RandomIdealGivenNorm` algorithm outputs a uniformly random primitive ideal, therefore the output of `GenIsogeny` is a uniformly random cyclic isogeny of degree $q(2^a - q)$ from E_{vk} .

Remark 4. There are different choices to encode the data defining the output of [Algorithm 1](#). The one above is the simplest one, and hence better for exposition. Other possibilities, more oriented towards efficiency and compactness and depending on the parameters, are discussed in [Section 7.2](#).

Algorithm 1 GenIsogeny(vk, sk, q)

Input: Secret key $\text{sk} = (E_{\text{vk}}, I_{\text{sk}}, M_{\text{sk}})$ and public key $\text{vk} = E_{\text{vk}}$, and prime number $q \in (2^{a-1}, 2^a)$.

Output: A two-dimensional representation of $\sigma : E_{\text{vk}} \rightarrow E_{\text{sig}}$ of degree $q(2^a - q)$.

- 1: $N \leftarrow q(2^a - q)$;
 - 2: $I_{\text{chall}} \leftarrow \text{RandomIdealGivenNorm}(\mathcal{O}_0, N)$;
 - 3: $I \leftarrow I_{\text{sk}} \cap I_{\text{chall}}$;
 - 4: $\varphi \leftarrow \text{IdealToIsogeny}(I)$;
 - 5: Set E_{sig} as the codomain of φ ;
 - 6: $t = q^{-1} \pmod{2^a}$
 - 7: $P_{\text{sig}} = [t \cdot m_{11}]\varphi(P_0) + [t \cdot m_{12}]\varphi(Q_0)$;
 - 8: $Q_{\text{sig}} = [t \cdot m_{21}]\varphi(P_0) + [t \cdot m_{22}]\varphi(Q_0)$;
 - 9: **return** $\text{iso} = (E_{\text{sig}}, P_{\text{sig}}, Q_{\text{sig}})$
-

Verification. We now define the verification algorithm $\text{VerIsogeny}(\text{iso}, E_{\text{vk}}, q)$. Parse the input iso as $(E_{\text{sig}}, P_{\text{sig}}, Q_{\text{sig}})$ where $P_{\text{sig}} = [q^{-1}]\sigma(P)$ and $Q_{\text{sig}} = [q^{-1}]\sigma(Q)$. We recall that $E_{\text{vk}}[2^a] = \langle P_{\text{vk}}, Q_{\text{vk}} \rangle$. We have to check that these points interpolate an isogeny $\sigma : E_{\text{vk}} \rightarrow E_{\text{sig}}$ of degree $q(2^a - q)$.

Write $\sigma = \varphi_q \circ \varphi_{q'} = \psi_{q'} \circ \psi_q$, where $\deg(\psi_q) = \deg(\varphi_q) = q$, and $\deg(\psi_{q'}) = \deg(\varphi_{q'}) = 2^a - q$. We can then factor σ using the commuting square

$$\begin{array}{ccc} E_{\text{vk}} & \xrightarrow{\varphi_{q'}} & E_2 \\ \downarrow \psi_q & & \downarrow \varphi_q \\ E_1 & \xrightarrow{\psi_{q'}} & E_{\text{sig}} \end{array}$$

This is a $(q, 2^a - q)$ -isogeny diamond, and therefore, by Kani's Lemma, the isogeny

$$\Phi : E_{\text{vk}} \times E_{\text{sig}} \rightarrow E_1 \times E_2$$

given by the matrix

$$\Phi = \begin{pmatrix} \psi_q & \widehat{\psi}_{q'} \\ -\varphi_{q'} & \widehat{\varphi}_q \end{pmatrix},$$

is a $(2^a, 2^a)$ -isogeny between these products of elliptic curves, viewed with their product polarisation. Furthermore, the kernel of Φ is given by

$$\ker(\Phi) = \{([q]P, \sigma(P)) \mid P \in E_{\text{vk}}[2^a]\} = \langle (P_{\text{vk}}, P_{\text{sig}}), (Q_{\text{vk}}, Q_{\text{sig}}) \rangle.$$

which is given to the verifier.

Given such a kernel, the verifier can attempt to compute Φ to verify that $(P_{\text{sig}}, Q_{\text{sig}})$ interpolates the isogeny σ . They first check that the codomain of Φ is an elliptic product, and reject the signature if not. Then, they can verify the degree of σ as follows. They compute $(P', _) := \Phi((P_{\text{vk}}, 0))$ and $(Q', _) := \Phi((Q_{\text{vk}}, 0))$. Observe that if $(E_{\text{sig}}, P_{\text{sig}}, Q_{\text{sig}})$ is a valid isogeny, then

$$e_{2^a}(P', Q') = e_{2^a}(P_{\text{vk}}, Q_{\text{vk}})^n,$$

where $n \in \{q, 2^a - q\}$ and e_{2^a} is the 2^a -Weil pairing. Therefore, it suffices to compute $e_{\text{vk}} = e_{2^a}(P_{\text{vk}}, Q_{\text{vk}})$ and $e' = e_{2^a}(P', Q')$ and check whether $e_{\text{vk}} = (e')^q$ or $e' = (e_{\text{vk}})^q$.

Hashing in Primes_a . For our signature scheme, we need to define the hash function H_{prime} that hashes into the set Primes_a . To construct H_{prime} we consider any cryptographic hash function $H_{a-2} : \{0, 1\}^* \rightarrow [0, 2^{a-2})$, composed with $h \mapsto 2^{a-1} + 2h + 1$ (prepending and appending a bit 1 to the binary expansion) to end up with an odd integer in the interval $(2^{a-1}, 2^a)$. Given E_{vk} and msg we define $H_{\text{prime}}(E_{\text{vk}} \parallel \text{msg})$ by repeatedly computing $2^{a-1} + 2H_{a-2}(E_{\text{vk}} \parallel \text{msg} \parallel \text{counter}) + 1$ for increasing values of counter until we hit a prime number. This requires on average $2^{a-2} / \#\text{Primes}_a \approx a \ln(2)/2$ repetitions. Notice that each step in the hash H_{prime} evaluation requires performing a primality test. In practice, we will use the Miller-Rabin test [52, 57].

The Miller-Rabin test is a probabilistic test that requires repeating an exponentiation by an a -bit exponent several times. The number of repetitions bounds the probability that a composite integer passes the primality test. Concretely, the higher the number of iterations performed, the lower this probability is. Recently, Cottaar, Galbraith, Maino and Trimoska estimated the number of iterations needed to have a secure instantiation of our construction [22]. To estimate the security of H_{prime} , we make the optimistic assumption that a full Miller-Rabin test has a quadratic cost in the bit length a . A more refined cost analysis of a Miller-Rabin test with k -iterations would take k exponentiations by an a -bit exponent. Even though this clearly exceeds the cost $O(a^2)$, such an optimistic estimate for our primality test will suffice.

Remark 5. A slightly faster choice for H_{prime} would be to first hash to a 2^a bits odd integer and then increase it sequentially until we reach a prime. However, this would introduce a bias towards primes associated to long intervals of non-prime integers, and we therefore avoid this option.

Remark 6. The value of counter can be provided to the verifier in Figure 4 at a minimal cost, its expected size being about $\log a$ bits, to save time during verification. However in this case a bit of care is needed, since different counters may lead to different primes. First of all, a verifier receiving a fixed counter that does not result in a prime hash should reject to prevent a sort of signature malleability (a valid signature with $\text{counter} - 1$ would also be valid). An upper

bound on `counter` should also be set to prevent an attacker to arbitrarily try many different primes for the same signature. For similar reasons, the concatenation inside `H` must ensure proper domain separation.

4 Security

In this section, we prove that both the identification protocol and the signature scheme are deeply connected to the hardness of evaluating large prime degree isogenies, a well understood problem in isogeny-based cryptography.

4.1 Key Recovery

First, note that the key recovery problem for both our constructions is simply the standard *Supersingular Endomorphism Ring Problem*, given below.

Problem 1 (Supersingular Endomorphism Ring Problem). Given a supersingular elliptic curve E defined over $\mathbb{F}_{p^2}$, find four (efficient representations of) endomorphisms which generate the ring $\text{End}(E)$.

Both the signing and the identification procedure result in revealing isogenies of large prime degree. These are hard to compute without the knowledge of the endomorphism ring. Moreover, it is believed that revealing such isogenies does not help to solve the endomorphism ring problem. This fact was first formulated in [24] where the authors argued that providing an oracle to produce isogenies of arbitrary degree would not impact the security of SQIsignHD. Furthermore, notice that given a curve E anyone can efficiently compute isogenies of large degree without knowledge of the endomorphism ring, as long as this degree is smooth. Since smooth-degree isogenies are sufficient to cover the whole isogeny graph, for each isogeny we reveal there exists an equivalent, smooth-degree isogeny that is computable without any knowledge of $\text{End}(E_{vk})$. This lends support to the assumption that our protocols do not leak useful information to an attacker.

4.2 Forgery and Impersonation

The security of the protocol relies on the following (new) assumption. Recall that Primes_a is the set of primes of exactly a bits. Consequently, an element $q \in \text{Primes}_a$ is uniquely determined by the value of $q(2^a - q)$.

Definition 5. A special degree isogeny oracle (*SPEDIO*) is an oracle which takes as input a supersingular elliptic curve E over $\mathbb{F}_{p^2}$ and a prime $q \in \text{Primes}_a$, and returns a uniformly random cyclic isogeny of degree $q(2^a - q)$ from E .

Problem 2. Given a random supersingular elliptic curve E and a SPEDIO, output an isogeny of degree $q'(2^a - q')$ with $q' \in \text{Primes}_a$ different from all degrees formerly generated by the oracle.

Remark 7. Using the same notation, the problem of computing a degree- q' isogeny reduces to solving [Problem 2](#). Indeed, given a degree $q'(2^a - q')$ isogeny, the degree q' component can be recovered in polynomial time by factoring it (as we do in the verification procedure). We note that the converse is not as straightforward, as there may be large prime factors in $(2^a - q')$ that are smaller than 2^{a-1} .

[Problem 2](#) can be summarized by the security game G_E^{pdeg} in [Figure 5](#), where E is a supersingular elliptic curve.

$G_E^{\text{pdeg}}(\mathcal{A})$:	$\text{SPEDIO}(q)$:
1: $\mathcal{Q} \leftarrow \emptyset$	1: assert $q \in \text{Primes}_a$;
2: $\sigma^* \leftarrow \mathcal{A}^{\text{SPEDIO}}(E)$	2: Sample isogeny $\sigma : E \rightarrow E'$ of de-
3: assert $\sigma^* : E \rightarrow E_{\text{sig}}^*$ is an isogeny	gree $q(2^a - q)$;
of degree $q(2^a - q)$	3: $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{q\}$;
4: assert $q \in \text{Primes}_a$	4: return $\sigma : E \rightarrow E_{\text{sig}}$
5: assert $q \notin \mathcal{Q}$	
6: return win	

Fig. 5. Security game for [Problem 2](#)

We now show the relation between the hardness of [Problem 2](#), the security against adaptive attacks of PRISM-id and the unforgeability of PRISM-sig.

Proposition 1. *Under the assumption that [Problem 2](#) is hard, any PPT adversary against adaptive attacks ([Definition 2](#)) of PRISM-id performing N interactions has a winning probability bounded by $N/\#\text{Primes}_a$.*

Proof. Let $\mathcal{A}$ be an adversary for the impersonator game in [Definition 2](#) with N interactions. Given the supersingular elliptic curve E , we simulate the impersonator game for $\mathcal{A}$ to win the game G_E^{pdeg} in [Figure 5](#), i.e., solve [Problem 2](#). We proceed as follows:

- During **Setup** we set $E_{\text{vk}} = E$ as public key and send it to $\mathcal{A}$;
- During the *probing phase* we use the oracle SPEDIO (defined in [Problem 2](#)) to perform N interactions with $\mathcal{A}$, let $\mathcal{Q}$ be the set of queried degrees, which is a set of size at most N . By [Remark 3](#), the output of SPEDIO has the same distribution as the response of an honest prover;
- In the *impersonation phase* the adversary $\mathcal{A}$ wins, i.e., $\text{output}_{\mathcal{A}} = \text{accept}$, if and only if $\mathcal{A}$ can provide an isogeny $\sigma : E \rightarrow E_{\text{sig}}$ of degree $q(2^a - q)$ for a uniformly random $q \in \text{Primes}_a$.

If $q \notin \mathcal{Q}$, the isogeny σ is a valid solution for G_E^{pdeg} , against the hardness of [Problem 2](#). Thus, the winning probability of $\mathcal{A}$ is bounded by

$$\Pr[q \in \mathcal{Q}] = \frac{\#\mathcal{Q}}{\#\text{Primes}_a} \leq \frac{N}{\#\text{Primes}_a}$$

as required. $\square$

We now establish the security of PRISM-sig in the standard model, leveraging only the collision resistance property of H_{prime} and the hardness of [Problem 2](#). Although this proof is inherently designed to simulate a signing procedure that returns a new random signature per each message msg , it can be straightforwardly adapted to always return the same isogeny when the same message is queried.

Proposition 2. *If H_{prime} is a collision-resistant cryptographic hash function and [Problem 2](#) is hard, then PRISM-sig is EUF-CMA secure ([Definition 4](#)).*

Proof. We show that given a PPT adversary $\mathcal{A}$ in the EUF-CMA model we can use it to win G_E^{pdeg} (that is equivalent to solving [Problem 2](#)) or find a collision for H_{prime} in polynomial time. Given the supersingular elliptic curve E , we set it as a public key E_{vk} in our signature scheme. For every message query msg_i , we evaluate the prime $q_i = H_{\text{prime}}(E_{\text{vk}} \parallel \text{msg}_i) \in \text{Primes}_a$ and query the oracle $\text{SPEDIO}(q_i)$ to get an isogeny $\sigma_i : E \rightarrow E_{\text{sig}}$ of degree $q_i(2^a - q_i)$ and we return it as a signature. By [Remark 3](#), these signatures follow the same distribution as honestly generated signatures for the public key E_{vk} . The strategy is described in [Figure 6](#). The set $\mathcal{M}$ contains the previously queried messages, while $\mathcal{Q}$ collects the values of H_{prime} applied to the elements in $\mathcal{M}$.

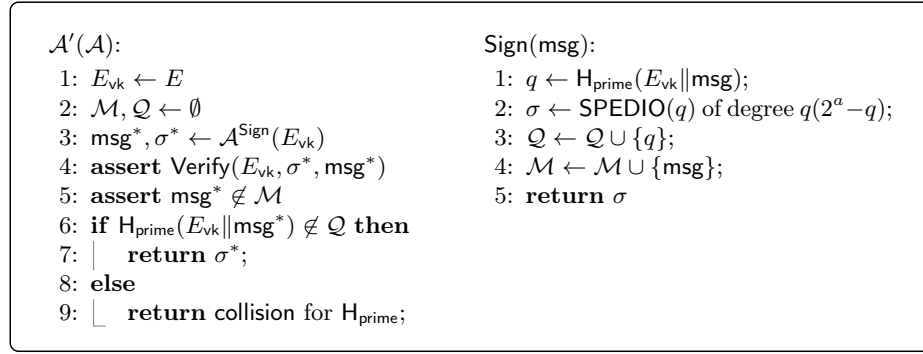


Fig. 6. Reduction from adversary $\mathcal{A}$ for PRISM-sig

Since σ^* is a valid signature (due to the assert in Line 4) it satisfies the assertions in Lines 3 and 4 from [Figure 5](#), i.e., it corresponds to a valid prime degree isogeny. We need only to check that it has not already been returned from SPEDIO before (see Line 5). If $q^* = H_{\text{prime}}(E_{\text{vk}} \parallel \text{msg}^*) \notin \mathcal{Q}$, then σ^* from Line 7 has a degree different from all isogenies previously returned. Indeed, recall that for all $q \in \text{Primes}_a$, we have $q > 2^a - q$, so for any $q \neq q^* \in \text{Primes}_a$, we have $q(2^a - q) \neq q^*(2^a - q^*)$. So, it is a valid solution to win G_E^{pdeg} .

On the other hand, if $H_{\text{prime}}(E_{\text{vk}} \parallel \text{msg}^*) \in \mathcal{Q}$ then in Line 9 we return a collision since we have found a message $\text{msg}_i \in \mathcal{M}$ such that $H_{\text{prime}}(E_{\text{vk}} \parallel \text{msg}_i) = H_{\text{prime}}(E_{\text{vk}} \parallel \text{msg}^*)$, but $\text{msg}_i \neq \text{msg}^*$. $\square$

4.3 Unforgeability in the ROM

If we model our hash function as a random oracle, we can give a security proof under a weaker hardness assumption. Namely, we can consider a variant of [Problem 2](#) in which the adversary has no control over the non-smooth isogenies being provided, nor in the degree of the output isogeny they forge. This problem is related to the hint-EndRing problem [2], which SQIsign relies on for its security; the two problems provide similar information to the adversary, but the hint-EndRing problem asks the adversary to compute the entire endomorphism ring, whereas our problem only asks the adversary to compute a non-smooth isogeny that they have not seen yet.

Problem 3. Given a random curve E , a set of N isogenies $\{\phi_i : E \rightarrow E_i\}_{i=1}^N$ of degree $q_i(2^a - q_i)$ for q_i uniformly random in Primes_a and ϕ_i uniformly random among the isogenies of degree $q_i(2^a - q_i)$, and a prime $\bar{q}$ uniformly random in $\text{Primes}_a \setminus \{q_i\}_{i=1}^N$, give an efficient representation of an isogeny of degree $\bar{q}(2^a - \bar{q})$.

It is immediate to see that if we can solve [Problem 3](#) then we can also solve [Problem 2](#), but potentially not vice-versa. [Remark 7](#) similarly applies to [Problem 3](#).

Proposition 3. *In the random oracle model (ROM), any PPT adversary that wins the EUF-CMA game ([Figure 2](#)) with advantage ϵ and that performs N_{sign} signing queries and N_H hashing queries can be used to solve [Problem 3](#) for $N = N_{\text{sign}} + N_H$ with probability at least ϵN_H^{-2} .*

Proof. Let $\mathcal{A}$ be a PPT adversary for the EUF-CMA of PRISM-sig. We want to define another PPT algorithm $\mathcal{B}$ that solves [Problem 3](#) for $N = N_{\text{sign}} + N_H$ using $\mathcal{A}$ as a subroutine in the random oracle model. For this $\mathcal{B}$ we need to simulate the answers to the signing and hashing queries. Let $\{\phi_i : E \rightarrow E_i\}_{i=1}^N$ be the isogenies of degree $q_i(2^a - q_i)$ for $q_i \in \text{Primes}_a$ received as input from [Problem 3](#), and $\bar{q}$ be the prime degree of the isogeny we have to find to solve [Problem 3](#).

Then, $\mathcal{B}$ fixes $E = E_{\text{vk}}$ and starts to simulate the interactions with $\mathcal{A}$ following the games G_0 and G_1 , as explained in [Figure 7](#). Since we are in the ROM, the adversary $\mathcal{A}$ needs to interact with $\mathcal{B}$ to query the hash function H_{prime} . Let msg^* be the output message of $\mathcal{A}$. We make the following assumptions on the queries:

1. Queries to H_{prime} are always of the form $E_{\text{vk}} \parallel \text{msg}$,
2. $E_{\text{vk}} \parallel \text{msg}^*$ is part of the N_H queries to H_{prime} , i.e., $\text{msg}^* \in \mathcal{H}$. This is because we can always modify $\mathcal{A}$ to query it before returning msg^* and the signature.
3. $\mathcal{H} \setminus \mathcal{M} \neq \emptyset$, i.e. there is always at least one message queried to H_{prime} but not to Sign. In fact by the previous point $\text{msg}^* \in \mathcal{H}$, but by the EUF-CMA definition we need $\text{msg}^* \notin \mathcal{S}$ to have non-zero winning probability, thus at least one element is in the set $\mathcal{H} \setminus \mathcal{M}$.

When looking at the game G_0 the only difference from G^{uf} ([Figure 2](#)), except for the output type, is the assertion in [Line 8](#). Since $j_{\bar{q}}$ is sampled independently

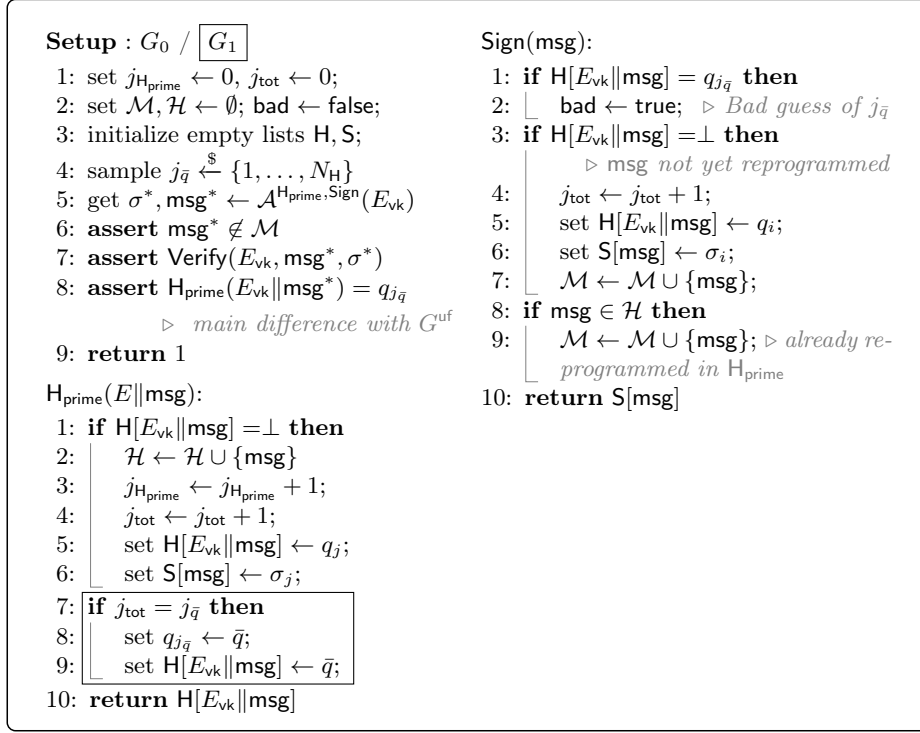


Fig. 7. Strategy to simulate the signing EUF-CMA model in the ROM

from all the randomness involved in the protocol and used by $\mathcal{A}$, we have that:

$$\begin{aligned}
 \Pr[G_0(\mathcal{A}) = 1] &= \Pr[G^{\text{uf}}(\mathcal{A}) = \text{win} \wedge \text{Hprime}(E_{\text{vk}} \parallel \text{msg}^*) = q_{j_{\bar{q}}}] = \\
 &= \Pr[G^{\text{uf}}(\mathcal{A}) = 1] \cdot \Pr[\text{Hprime}(E_{\text{vk}} \parallel \text{msg}^*) = q_{j_{\bar{q}}}] = \epsilon \cdot \frac{1}{N_{\text{H}}} . \quad (5)
 \end{aligned}$$

Games G_0 and G_1 are identical until the **bad** flag is set to **true**, since $q_{j_{\bar{q}}}$ and $\bar{q}$ have the same distribution. So, the difference is relevant only in the case of a sign query on the same message. Hence, by standard game-based proof results [3, Lemma 3.7] we have that

$$\Pr[G_0(\mathcal{A}) = 1 \wedge \text{Good}_0] = \Pr[G_1(\mathcal{A}) = \text{win} \wedge \text{Good}_1], \quad (6)$$

where Good_i is the event that the flag **bad** is never changed in the game G_i . Now, observe that in game G_0 there is no reprogramming to $\bar{q}$, thus the two events in the left hand side of Equation (6) are independent. Moreover, as argued before, there is at least one message $\mathcal{H} \setminus \mathcal{M}$, i.e., a message for which G_0 reprograms Hprime but never queries to **Sign**. Since $j_{\bar{q}}$ is chosen independently of the randomness used by $\mathcal{A}$ there is at least a probability $1/N_{\text{H}}$ that the reprogrammed message

is in $\mathcal{H} \setminus \mathcal{M}$, that implies Good_0 . Combining everything we have

$$\Pr[G_1(\mathcal{A}) = 1 \wedge \text{Good}_1] \stackrel{(6)}{=} \Pr[G_0(\mathcal{A}) = \text{win}] \Pr[\text{Good}_0] \stackrel{(5)}{\geq} \epsilon \frac{1}{N_H^2}.$$

It is clear that if $G_1(\mathcal{A}) = 1$, the isogeny σ^* from Line 5 is a valid solution for Problem 3 that $\mathcal{B}$ can return. $\square$

Remark 8. This reduction strongly relies on the signature being deterministic, as a function of the message msg and the public key E_{vk} . In fact, providing two different signatures for the same message msg requires returning two isogenies of the same degree $H_{\text{prime}}(E_{\text{vk}} \parallel \text{msg})$, but the initial data given in Problem 3 only provides us with one isogeny per prime number q_i .

4.4 Best Known Attacks on Hardness Assumptions

In this section we discuss the best known attacks against the conjectured hard problems underlying our scheme. This analysis will guide our choice of parameters.

Endomorphism ring problem. As argued above, key recovery against both PRISM-sig and PRISM-id amounts to the computation of the endomorphism ring of the public key E_{vk} . The discussion in [24, Sec. 4] justifies the assumption that E_{vk} is a random curve. Thus, the fastest known algorithms to compute its endomorphism ring have classical complexity in $\tilde{O}(p^{1/2})$ [30]. The only known quantum speed-up uses Grover’s algorithm [41, 14], achieving a quantum complexity of $\tilde{O}(p^{1/4})$. These are the best key recovery attacks against most isogeny-based protocols, including the SQIsign family (see e.g. [9, Sec. 6]). We hence require p to be a prime of around 2λ bits for a security level of λ bits.

Computing isogenies of prime degree. The most direct attempt to solve Problems 2 and 3 requires, on input of a prime degree q , to compute a cyclic degree $q(2^a - q)$ isogeny from E_{vk} without the knowledge of $\text{End}(E_{\text{vk}})$. First of all, notice that $q \in \text{Primes}_a$, i.e., it is a prime bigger than 2^{a-1} , and so q is the biggest prime factor of $q(2^a - q)$. As a consequence, the cost of computing a $q(2^a - q)$ -isogeny will mostly depend on the cost of computing a degree q isogeny. In fact, an attacker may even compute isogenies whose degrees are pairwise coprime factors of $q(2^a - q)$ in parallel, all starting from E_{vk} , and then compose them by pushing them forward [60, Prop. 6.15]. The complexity of this approach is dominated by that of computing an isogeny of largest prime power degree. We can thus restrict to studying the cost of computing an isogeny of (large) prime degree q .

There are various methods to compute q -isogenies without relying on the knowledge of the endomorphism ring. An approach is to use Vélú’s formulae [65]. These formulae require knowledge of a point of order q . In general, such a point will not be defined over $\mathbb{F}_{p^2}$, but rather over a large field extension. Specifically, we

expect this degree to be roughly the size of q [36]. Field operations in an extension of degree q have an overhead that is in $\tilde{O}(q)$. The amount of field operations needed to compute a degree- q isogeny using Vélu's formulae is again linear in q , and so we obtain a total complexity of $\tilde{O}(q^2)$. A significant improvement in computing an isogeny from its kernel is achieved through the square-root Vélu algorithm [11]. This algorithm reduces the number of field operations from $\tilde{O}(q)$ to $\tilde{O}(q^{1/2})$. As this computation runs over the field extension where a point of order q is defined, the resulting expected complexity of using square-root Vélu algorithm is $\tilde{O}(q^{3/2})$. However, this complexity assumes that a point of order q is already available, and we must still factor in this cost. One method for obtaining such a point is to sample a random point on the curve $E/\mathbb{F}_{p^q}$ and multiply it by the appropriate cofactor of size around $\frac{p^q}{q}$, by Hasse's Theorem. This requires at least $\log\left(\frac{p^q}{q}\right) \approx q$ point doublings and additions defined on a field extension of degree q , and thus has complexity $\tilde{O}(q^2)$. Another option is to instead search for its x -coordinate as a root of the q -division polynomial of degree $(q^2 - 1)/2$. This also requires at least $\tilde{O}(q^2)$ field operations.

A different approach to computing q -isogenies avoiding large field extensions is to employ kernel polynomials. Once a kernel polynomial has been computed, it is possible to use well-known formulae, such as those in [46], to compute the q -isogeny with $O(q)$ field operations. Obtaining kernel polynomials without access to an isogeny representation or a point of order q (defined over a large field extension) is also a costly operation. To the best of our knowledge, the fastest method to compute the kernel polynomial is via Elkies algorithm [35] (see, for example, [39, Chapter 25.2.1]). There are two costly steps to this algorithm. First, we need to find the root of the modular polynomial $\Phi_q(X, Y)$ over $\mathbb{F}_p$, costing $O(q)$ operations, assuming the modular polynomials has already been computed. Secondly, we need to compute q coefficients via a recurrence relation involving all previous coefficients, requiring $\mathcal{O}(q^2)$ operations in $\mathbb{F}_p$.

Another method is to factor the q -division polynomial, and extract the kernel polynomial. However, this yields a complexity much larger than the method discussed above.

Remark 9. A more careful analysis of the above approach is carried out in [53]. If $q \mid ((-p)^k - 1)$, points of order q are defined over $\mathbb{F}_{p^{2k}}$ and the above strategy has complexity $\tilde{O}(k^2)$. Thus an attacker can fix k in advance, and for each k compute its success probability (i.e., the chance that a random q divides $((-p)^k - 1)$) as well as its running time. They conclude that $a = 7\lambda/6$ is in any case sufficient to guarantee security. In PRISM-sig, a must already be bigger than that to guarantee the security of the hash function, as discussed below. For PRISM-id in principle one could use $a = \lambda + \log \lambda$ to guarantee that the set of primes is big enough, which falls slightly short of this bound. However, this attack is easily prevented by sampling q as a *safe prime*, i.e., a prime such that $q - 1 = 2q'$ with q' also a prime (q' is then called a *Sophie Germain prime*); $a = \lambda + 2 \log \lambda$ still ensures that there are at least 2^λ valid a -bit primes.

The quantum complexity of computing prime degree isogenies has never been thoroughly studied. However, we see no reason to believe that this problem would be amenable to a significant quantum speed-up.

To conclude, when targeting λ -bits security, to guarantee the security of [Problems 2 and 3](#) we only have to impose $a = \tilde{O}(\lambda)$. This is easily achievable since we already imposed $p \approx 2^{2\lambda}$ to prevent endomorphism ring computations, and hence have 2λ bits of 2-torsion to spare. These are the only attacks against PRISM-id we are aware of. On the other hand, in the construction of PRISM-sig, we have to take into account the security of the hash function, which turns out to be the actual security bottleneck, as argued below.

Remark 10. We highlight that, when $q^2 > p^{1/2}$, it is more efficient to compute the endomorphism ring of the starting curve; such a computation takes $\tilde{O}(p^{1/2})$ base field operations. Thus, computing a q -isogeny originating from a curve E of unknown endomorphism ring requires $\tilde{O}(\min(q^2, p^{1/2}))$ base field operations.

Breaking the hash function. First of all, to avoid signature reusing, i.e., to achieve the non-resignability property from [\[23\]](#), we insert the domain isogeny E_{vk} into the hashing input, since this comes at virtually no cost. Then, as our construction PRISM-sig follows a hash-and-sign paradigm, an attacker can obtain forgeries by finding collisions for H_{prime} .

A priori, it may seem that hashing into Primes_a is not enough to achieve λ -bit security against collision search. Indeed, there are only about $2^{a-1}/(a \ln(2))$ primes in Primes_a . However, we do not hash directly into Primes_a , but rather into the set of odd integers in $(2^{a-1}, 2^a)$ via H_{a-2} and $h \mapsto 2^{a-1} + 2h + 1$, and reject until we find a prime. Recall that the expected number of tries is about $a \ln(2)/2$. Also, evaluating H_{prime} not only requires evaluating multiple times H_{a-2} , but also performing primality tests. Under the mild assumption that the primality test has a quadratic cost in the bit length of the number being tested, which is a , and that H_{a-2} has a cost linear in a , we have that each hash evaluation has an average evaluation cost cubic in a . This means that a birthday-bound like attack finding collisions via repeated evaluations of H_{prime} has a cost of $O(a^3 \sqrt{\#\text{Primes}_a})$, above the 2^λ threshold even for $a < 2\lambda$.

To set parameters for H_{prime} collision resistance we consider a more sophisticated attack. First, the attacker finds multiple collisions for H_{a-2} until one of these collisions is a prime collision. This makes the cost of primality testing negligible and leverages the fact that from a large list of random values it is easier to find multiple collisions than find a single collision for H_{prime} . Precisely, from a list of t random strings of $a - 2$ bits, the expected number of pair collisions is $\binom{t}{2} 2^{-a+2}$.¹¹

Thus, in order to produce a collision for H_{prime} , the expected number t of calls to H_{a-2} satisfies

$$\binom{t}{2} 2^{-a+2} \approx a \ln(2)/2.$$

¹¹ there are $\binom{t}{2}$ pairs and each of them is a collision with probability 2^{-a+2} .

The same estimate can be obtained by the algorithm proposed in [64, Sec. 4] to find good collisions. Thus, to get λ -bits collision resistance we want that

$$2^\lambda \leq t \approx 2^{\frac{a-2}{2}} \sqrt{a}, \quad (7)$$

leading to the asymptotic estimate $a \approx 2\lambda - \log \lambda$. Notice that this requirement is much stronger than $a > \frac{1}{2}\lambda$ resulting from the discussion above. On the other hand, here we have just attributed a cost of 1 to each call to H_{a-2} , which is a very conservative choice. We leave it to the reader to take into account more realistic cost estimates, where one could even use artificially slow hash functions to establish a further reduction of a (but this also affects signing and verification). This trick is reminiscent of other schemes that employ slow hashing or *proof-of-work* [10,13].

It is clear that $a = e \approx 2\lambda$ is good enough to get collision resistance, but we can do slightly better by taking the smallest a satisfying Equation (7). For example, for NIST Level I security (namely, $\lambda = 128$) we can take $a = 251$. Note: in our concrete parameters from Section 7.1 we will choose $a = 248$, but this defect is (amply) compensated by the complexity of evaluating H_{a-2} .

Finally, as was already remarked, given two isogenies of coprime degree from a curve E there exists a polynomial time algorithm (that eventually requires to go to dimension four or eight) to compute the respective pushforwards. This implies that an attacker seeing an isogeny of degree $q(2^a - q)$ can effectively obtain an isogeny for all prime power factors of $2^a - q$ that can then be reused later: the output of the hash function just consists of q . This also explains why we require q to have exactly a bits. Indeed, in this way we ensure that none of the factors of $2^a - q$ will land in Primes_a , and thus by seeing them an attacker does not learn anything.

5 Towards a strongly unforgeable variant of PRISM

A desirable property for a digital signature scheme to have is strong unforgeability (Definition 4), meaning it should not be possible to forge a signature even for an already signed message. This stronger security notion provides additional security guarantees, which are relevant for some constructions (see e.g., [4]). It is also plausible that advanced schemes relying on PRISM would require such a property. Indeed, in applications like cryptocurrencies, such malleability can lead to exploitable vulnerabilities [5,29,49].

It is clear that PRISM-sig does not achieve this. Indeed, suppose that we are given a signature isogeny $\sigma : E_{vk} \rightarrow E_{sig}$ corresponding to a message msg such that $\deg(\sigma) = q(2^a - q)$, $q \in \text{Primes}_a$ and $2^a - q$ is divisible by a small prime ℓ . Such a signature isogeny can be written as $\sigma = \psi_\ell \circ \rho$, where $\deg(\psi_\ell) = \ell$ and $\deg(\rho) = q(2^a - q)/\ell$. Then, we can construct a distinct signature isogeny σ' for the same message msg by composing the isogeny ρ with an isogeny $\psi'_\ell \neq \psi_\ell$ of degree ℓ . In particular, computing such an isogeny ψ'_ℓ does not require the knowledge of $\text{End}(E_{vk})$ since ℓ is a small prime.

The most natural way to counter this attack is to only consider signature isogenies whose degree is prime. In what follows, we describe a variant of PRISM-sig satisfying this requirement.

In this new variant, the **KeyGen** algorithm is left unchanged. The same does not hold for **Sign** and **Verify**. For this reason, we modify some of the building blocks in PRISM.

We first define the set Squares_e of all the integers s such that $2^{2e-1} - s^2$ is a prime;¹² to be more precise

$$\text{Squares}_e := \{0 < s < 2^{e-1} \mid 2^{2e-1} - s^2 \text{ is prime}\}.$$

We replace the hash function H_{prime} with the hash function $H_{\text{squares}} : \{0, 1\}^* \rightarrow \text{Squares}_e$ that outputs odd integers s such that $2^{2e-1} - s^2$ is prime. Similarly to H_{prime} , the hash function H_{squares} samples odd integers s in $(0, 2^{e-1})$ iterating H_{e-2} as $h \mapsto 2h + 1$ until $2^{2e-1} - s^2$ is prime.

We tweak the algorithm **GenIsogeny** to output a unique representation of an isogeny of prime degree. More precisely, we define **GenIsogenyStrong** in [Algorithm 2](#). To ensure uniqueness of the output, we introduce the relation $<_{\mathbb{F}_{p^2}}$ over $\mathbb{F}_{p^2}$: given two distinct $\alpha = a_1 + ia_2$ and $\beta = b_1 + ib_2$ in $\mathbb{F}_{p^2}$, we say that $\alpha <_{\mathbb{F}_{p^2}} \beta$ if and only if either $a_1 < b_1$ or $a_1 = b_1$ and $a_2 < b_2$.

Algorithm 2 GenIsogenyStrong(vk, sk, s)

Input: Secret key $\text{sk} = (E_{\text{vk}}, I_{\text{sk}}, M_{\text{sk}} = (\begin{smallmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{smallmatrix}))$, a public key $\text{vk} = E_{\text{vk}}$, and an odd integer $0 < s < 2^{e-1}$.

Output: A two-dimensional representation of $\sigma : E_{\text{vk}} \rightarrow E_{\text{sig}}$ of degree $2^{2e-1} - s^2$.

- 1: $I_{\text{chall}} \leftarrow \text{RandomIdealGivenNorm}(\mathcal{O}_0, 2^{2e-1} - s^2)$;
 - 2: $I \leftarrow I_{\text{sk}} \cap I_{\text{chall}}$;
 - 3: $\varphi \leftarrow \text{IdealToIsogeny}(I)$;
 - 4: Set E_{sig} as the codomain of φ ;
 - 5: $t = s^{-1} \pmod{2^e}$
 - 6: $P_{\text{sig}} = [t \cdot m_{11}] \varphi(P_0) + [t \cdot m_{12}] \varphi(Q_0)$;
 - 7: $Q_{\text{sig}} = [t \cdot m_{21}] \varphi(P_0) + [t \cdot m_{22}] \varphi(Q_0)$;
 - 8: **if** $y(-P_{\text{sig}}) <_{\mathbb{F}_{p^2}} y(P_{\text{sig}})$ **then**
 - 9: $(P_{\text{sig}}, Q_{\text{sig}}) \leftarrow (-P_{\text{sig}}, -Q_{\text{sig}})$
 - 10: **return** $(E_{\text{sig}}, P_{\text{sig}}, Q_{\text{sig}})$
-

The signing algorithm is analogous to the one in PRISM-sig where H_{squares} replaces H_{PRISM} and **GenIsogenyStrong** replaces **GenIsogeny**. We now focus on **Verify**.

Given a representation of $\sigma : E_{\text{vk}} \rightarrow E_{\text{sig}}$ of degree $2^{2e-1} - s^2$, it is possible to embed σ into the two-dimensional isogeny Φ whose matrix form is given as

¹² Considering the exponent $2e - 1$ is rather a natural choice. In fact, having an even exponent such as $2e$ prevents us from finding primes since $2^{2e} - s^2 = (2^e - s)(2^e + s)$.

$$\Phi = \begin{pmatrix} [s] & -\widehat{\sigma} \\ \sigma & [s] \end{pmatrix} : E_{\text{vk}} \times E_{\text{sig}} \rightarrow E_{\text{vk}} \times E_{\text{sig}}.$$

We can write $\Phi = \widehat{\Phi}_2 \circ \Phi_1 : E_{\text{vk}} \times E_{\text{sig}} \rightarrow E_{\text{vk}} \times E_{\text{sig}}$, where $\deg(\Phi_1) = 2^e$ and $\deg(\Phi_2) = 2^{e-1}$. In particular

$$\ker(\Phi_1) = \{([s]P, -\sigma(P)) \mid P \in E_{\text{vk}}[2^e]\}$$

and

$$\ker(\Phi_2) = \{([s]P, \sigma(P)) \mid P \in E_{\text{vk}}[2^{e-1}]\}.$$

This means that we can represent σ using only the image of σ on the 2^e -torsion. If $\Phi_1 = \Psi_e \circ \Psi_{e-1} \cdots \circ \Psi_1$, then $\Phi_2 = \Psi_{e-1} \cdots \Psi_1 \circ \iota$, where $\iota(P, Q) = (P, -Q)$. In other words $\Phi_1 = \Psi_e \circ \Phi_2 \circ \iota$. Hence, to represent σ , it suffices to compute $\Psi_1, \dots, \Psi_e$.

The special structure of Φ allows for an alternative way to check that Φ embeds σ without relying on any pairing computations. In particular Φ embeds an isogeny of degree $2^{2e-1} - s^2$ if and only the top-left component of Φ is the scalar multiplication by s . To check that the top-left component of Φ is the scalar multiplication by s , we need to prove that, for all the points $T \in E_{\text{vk}}[N]$ where $N^2 > 2^{2e} + 1 > 4s^2 + 1$, we have

$$\Phi \begin{pmatrix} T \\ 0 \end{pmatrix} = \begin{pmatrix} [s]T \\ 0 \end{pmatrix}.$$

Equivalently, composing both sides with Φ_2 , we have to prove that

$$\Psi_e \circ \Phi_2 \begin{pmatrix} [2^{e-1}]T \\ 0 \end{pmatrix} = \Phi_2 \begin{pmatrix} [s]T \\ 0 \end{pmatrix}.$$

Since the characteristic p of the base field is equal to $c \cdot 2^e - 1$ and $c > 1$, we can take $N = p + 1$. We summarize the verification procedure in the algorithm `VerIsogenyStrong` ([Algorithm 3](#)).

The verification algorithm is analogous to the one in `PRISM-sig` where `VerIsogenyStrong` replaces `VerIsogeny`.

5.1 Security

The strong unforgeability of this variant can be proven following the same proof strategies as in [Section 4.2](#) and [4.3](#). However, we need to rely on stronger hardness assumptions.

To prove strong unforgeability in the standard model, we introduce a stronger version of [Problem 2](#), and then we would need to adjust the proof in [Proposition 2](#). Similarly to [Problem 2](#), we require an oracle that computes isogenies of the correct degree; we call this oracle `SPEDIO`².

Algorithm 3 VerIsogenyStrong($vk, E_{\text{sig}}, P_{\text{sig}}, Q_{\text{sig}}, s$)

Input: The public key $vk = E_{vk}$, the output $(E_{\text{sig}}, P_{\text{sig}}, Q_{\text{sig}})$ of GenIsogeny' (Algorithm 2) and an odd integer $0 < s < 2^{e-1}$.

Output: True or False.

```
1: if  $y(P_{\text{sig}}) <_{\mathbb{F}_{p^2}} y(-P_{\text{sig}})$  then
2:   return False
3: Compute  $(P_{vk}, Q_{vk}) = E_{vk}[2^e]$ .
4: Try to compute the isogeny  $\Phi$  with kernel  $\langle ([2]P_{vk}, [2]P_{\text{sig}}), ([2]Q_{vk}, [2]Q_{\text{sig}}) \rangle$ .
5: Try to compute the isogeny  $\Psi$  with kernel  $\langle \Phi(P_{vk}, -P_{\text{sig}}), \Phi(Q_{vk}, -Q_{\text{sig}}) \rangle$ .
6: Compute a basis  $(R, T)$  of  $E_{vk}[p+1]$ .
7:  $(R', T') \leftarrow \Phi(R, 0), \Phi(T, 0)$ 
8: if  $[2^{e-1}]\Psi(R') = [s]R'$  and  $[2^{e-1}]\Psi(T') = [s]T'$  then
9:   return True
10: else
11:   return False
```

Definition 6. A special squared degree isogeny oracle (SPEDIO²) is an oracle that takes as input a supersingular elliptic curve E over $\mathbb{F}_{p^2}$ and an $s \in \text{Squares}_e$, and returns a uniformly random cyclic isogeny of degree $2^{2e-1} - s^2$ originating from E .

Problem 4. Given a random supersingular elliptic curve E and a SPEDIO², output an isogeny, different from all the isogenies formerly generated by the oracle, of degree $2^{2e-1} - \bar{s}^2$ with $\bar{s} \in \text{Squares}_e$.

Similarly, to prove strong unforgeability in the ROM, we would need to rely on a stronger variant of Problem 3 and adjust the proof in Proposition 3. Analogously to Remark 8, we have to impose that signatures are deterministically generated for a given degree.

Problem 5. Given a random curve E , a set of N isogenies $\text{Iso} := \{\phi_i : E \rightarrow E_i\}_{i=1}^N$ of degree $2^{2e-1} - s^2$ for s uniformly random in Squares_e and ϕ_i uniformly random among the isogenies of degree $2^{2e-1} - s^2$, and an $\bar{s}$ uniformly random in Squares_e , give an efficient representation of an isogeny of degree $2^{2e-1} - \bar{s}^2$ not in Iso .

The main difference between Problems 4 and 5 and Problems 2 and 3 is that it is possible to break the former with an isogeny having the same degree as the ones computed by SPEDIO² or previously given. This relaxation does not seem to make the problem of computing non-smooth degree isogenies any easier. Indeed, a similar problem has already appeared in the literature [50, Problem 1]. Therefore we believe that the security analysis conducted in Section 4.4 still holds.

On the security of H_{squares} . The security of this strongly-unforgeable variant also depends on the robustness of H_{squares} . In this paragraph, we argue that,

assuming the validity of a well-known conjecture, the same arguments for collision resistance of H_{PRISM} apply to H_{squares} .

First, we explain why we believe that H_{squares} outputs a prime of the form $2^{2e-1} - s^2$ with probability $\approx \frac{\sqrt{2} \ln((e-1) \ln(2))}{(e-1) \ln(2)}$. The distribution of such primes is connected to one of Hardy-Littlewood conjectures, namely [43, Conjecture H]. Hardy and Littlewood conjectured that every large, non-square integer may be expressed as the sum of a prime and a square. Additionally, they speculate that for a large integer n , the number of pairs consisting of a prime and a square summing up to n should be asymptotically equal to

$$\frac{\sqrt{n}}{\ln(n)} \prod_{\ell \geq 3, \ell \text{ is prime}} \left(1 - \frac{\left(\frac{n}{\ell}\right)}{\ell-1}\right),$$

which, under the generalized Riemann hypothesis, can be upperbounded with $\frac{\sqrt{n} \ln \ln(n)}{\ln(n)}$. Hence, assuming that such a conjecture holds, our claim on the

probability of hashing into a prime being $\approx \frac{\sqrt{2} \ln((e-1) \ln(2))}{(e-1) \ln(2)}$ holds.

Now, we show why the probability of obtaining a collision for H_{squares} is negligible. Following the same reasoning as in the subsection “Breaking the hash function” of Section 4.4, we have that in order to produce a collision for H_{squares} , the expected number t of calls to H_{e-2} satisfies

$$\binom{t}{2} 2^{-e+2} \approx \frac{\sqrt{2} \ln((e-1) \ln(2))}{(e-1) \ln(2)}.$$

This implies that $t \approx 2^{\frac{e-2}{2}} \sqrt{e-1}$, which is clearly $\geq 2^\lambda$. Thus, the probability of obtaining a collision for H_{squares} is negligible in λ .

5.2 Comparison with a four-dimensional version

At the beginning of Section 5, we explained that one way to obtain strong unforgeability is to consider signature isogenies of prime degree. Then, we proceeded to describe a variant of PRISM-sig whose signature isogenies can be represented via two-dimensional isogenies. However, one may wonder why we did not simply consider using four-dimensional isogenies to represent signatures and leave the rest of the construction unchanged. In this section, we justify our design choices comparing them with a variant where signature isogenies are represented via four-dimensional isogenies.

The first reason to rely on two-dimensional isogenies may be simply dictated by the fact that two-dimensional isogenies are less costly than four-dimensional ones. While this is clearly true, we also have to take into account that our two-dimensional variant requires computing a $(2^{2e-1}, 2^{2e-1})$ -isogeny whereas a four-dimensional variant would require computing a $(2^a, 2^a, 2^a, 2^a)$ -isogeny, where $2\lambda - \log \lambda \approx a < e \approx 2\lambda$. Nevertheless, as we showed above, the $(2^{2e-1}, 2^{2e-1})$ -isogeny utilized in our construction can be computed by only using a $(2^e, 2^e)$ -isogeny, which makes our design more efficient.

Another reason to utilize our variant is that `VerIsogenyStrong` (Algorithm 3) does not use pairings. It suffices to evaluate a $(2^e, 2^e)$ -isogeny on the $(p+1)$ -torsion and check for some equalities.

Additionally, we discuss one subtlety that we would have encountered when hashing into primes in the case of the four-dimensional variant. While it may seem that we could have used H_{PRISM} to output a prime q determining the degree of the signature isogeny, we remark that, to represent a q -isogeny in dimension four, we must also ensure that $2^a - q$ can be expressed as a sum of two squares. In practice, this would have required to search for primes q such that $2^a - q$ is a prime congruent to 1 mod 4. Finding such primes is connected to the extended Goldbach conjecture, for which heuristics similar to the ones described above exist. Hence, we believe that hashing into primes q such that $2^a - q$ is a prime equal to 1 modulo 4 has roughly the same cost as hashing into primes of the form $2^{2e-1} - s^2$.

Finally, we remark that comparing Problems 4 and 5 to the hardness assumption we would use in the four-dimensional variant is difficult and, to the best of our knowledge, there is no reason to believe that either assumption is any easier. In the case of the four-dimensional variant, we would have had access to isogenies of prime degree q such that $2^a - q$ is a prime equal to 1 modulo 4. In Problems 4 and 5, we have isogenies of prime degree $2^{2e-1} - s^2$. The degrees of the isogenies in Problems 4 and 5 are generally larger than the ones used for the four-dimensional version. Currently, it is unclear how to exploit this difference.

Taking everything into consideration, we argue that our design choices are more efficient than using signature isogenies represented in dimension four, while we see no reason why these choices would imply lower security guarantees.

6 salt-PRISM: a salted variant of PRISM

Recall from Section 4.4 that the main limiting factor to the size of a (and hence to the verification time of PRISM-sig) is the security of the hash function against collision attacks. In this section, we present a modified version of protocol PRISM-sig in which a random salt is added to the message before hashing. This salt is then published alongside the signature. The modification relies on domain separation of the hash function $H : \text{Ell}_p \times \{0, 1\}^* \times \{0, 1\}^k \rightarrow \{0, 1\}^n$, as done in, for example [42], to mitigate offline collision attacks. We identify n -bit digests with integers in $[0, 2^n)$, for $n \geq a$ to ensure that Primes_a is contained in the image of H . With this modification the adversary cannot predict the salt that will be used for signing a message, and thus cannot precompute collisions of a signed message, instead having to resort in finding second preimage collisions for the hash function, which is a much harder problem. This will allow us to provide a rigorous security proof in the (Q)ROM that we use to decrease the value of a needed for our signatures, resulting in better efficiency without compromising security. We exhibit this variant, called salt-PRISM, in Figure 8.

Signer	Verifier
$pp = (p, \text{Primes}_a, H : \text{Ell}_p \times \{0, 1\}^* \times \{0, 1\}^k \rightarrow \{0, 1\}^n, B_{\text{salt}})$	
$sk = \phi_{sk}$	$vk = E_{vk}$
repeat $\text{salt} \leftarrow \{0, 1\}^k$ $q \leftarrow H(E_{vk}, \text{msg}, \text{salt})$ until $q \in \text{Primes}_a$ $\sigma \leftarrow \text{GenIsogeny}(vk, sk, q)$	
$\xrightarrow{\text{msg, salt, } \sigma}$	
$q \leftarrow H(E_{vk}, \text{msg}, \text{salt})$ if $q \in \text{Primes}_a$ and $\text{VerIsogeny}(\sigma, E_{vk}, q)$ return accept	

Fig. 8. salt-PRISM signature scheme, with additional salting.

6.1 Security of salt-PRISM

Finding a proof for the security of salt-PRISM in the (Q)ROM is quite straightforward by adapting the techniques from [40] and the *Measure-and-Reprogram Technique* from [32], as used in [48]. However, there are some subtleties to take into account. First, we cannot simulate signing queries ourselves, as we need to get them as input from Problem 3. Moreover, we need to hash into primes, i.e., to a subset of the output space of the random oracle. This creates a bias in the distribution of the salt.

To deal with the latter issue, we:

- Use the *Adaptive Reprogramming Theorem* from [40] that we specialize to Corollary 1 below.
- Define a new variant of the *Multi-target Extended target collision resistance* problem, called *biased Multi-target Extended target collision resistance* (b-eTCR), in Lemma 1 below. We also generalize known unconditional results [40,44] against this problem using idealized hash functions in Equation (9).

Corollary 1 (of Theorem 1 from [40]). *Let X_1, X_2, Y' and Y be some finite sets with $Y' \subset Y$, and let $\gamma = |Y'|/|Y|$. Let $\mathcal{D}$ be any distinguisher for the game in Figure 9, issuing q many (quantum) queries to H , with $q \leq \gamma|X_1|$, and R many reprogramming instructions in the form of a value x_2 . Then:*

$$|\Pr[\text{REPRO}_1^{\mathcal{D}} \rightarrow 1] - \Pr[\text{REPRO}_0^{\mathcal{D}} \rightarrow 1]| \leq \frac{3R}{2} \sqrt{\frac{q}{\gamma|X_1|}}. \quad (8)$$

Proof. We use the notation from the main theorem in [40]. The corollary follows via noting that at the r -th query to the reprogramming oracle, REPROGRAM is sampling x_1 uniformly from the set of x_2 such that $H(x_1, x_2) \in Y'$. Since H_0 is a uniform random function the expected size of this set is $\gamma|X_1|$, thus the expected

value for $p_{\max}^{(r)} = \mathbb{E} \max_{x_1} \Pr(x_1)$ is $p = 1/(\gamma|X_1|)$ (independent of r). In the main theorem from [40] the expectation is over the behavior of the distinguisher up to the r -th query, that includes also the choice of the random oracle H_0 . The bound in (8) then follows by the bound in Theorem 1 from [40], via upper bounding $\hat{q}_r$ with q for each r and qp (< 1 by assumption) with $\sqrt{qp}$. $\square$

REPRO_b($\mathcal{D}$)	REPROGRAM(x_2):
1: $H_0 \leftarrow_{\$} Y^{X_1, X_2}$	1: repeat $(x_1, y) \leftarrow X_1 \times Y$
2: $H_1 := H_0$	2: until $H(x_1, x_2) \in Y'$
3: $b' \leftarrow \mathcal{D}^{H_b}, \text{REPROGRAM}$	3: $H_1 := H_1^{(x_1, x_2) \mapsto y}$
	4: return x_1

Fig. 9. Adaptive reprogramming games REPRO_b for $b \in \{0, 1\}$.

Lemma 1. *Let X_1, X_2, Y' and Y be some finite sets with $Y' \subset Y$, and let $\gamma = |Y'|/|Y|$. Let $\mathcal{A}$ be a (quantum) adversary against the game from Figure 10 issuing at most q_b queries to the Box oracle and q_h queries to the random oracle H . Then, for any adversary $\mathcal{A}$ performing at most q_h classical queries to H :*

$$\text{Adv}(\mathcal{A}) \leq \frac{(q_h + 1) \cdot q_b}{|Y|} + \frac{q_h q_b}{\gamma |X_1|}. \quad (9)$$

For any adversary $\mathcal{A}$ performing at most q_h quantum queries to H :

$$\text{Adv}(\mathcal{A}) \leq \frac{8q_b(q_b + q_h + 2)^2}{|Y|} + \frac{3q_b}{2} \sqrt{\frac{q_h + q_b + 1}{\gamma |X_1|}}. \quad (10)$$

Proof. We adapt the same proof blueprint as in [44, 40].

Game G_0 : In this game we consider a modification of the reduction as in [44, Figure 2]. We consider the one query case $q_b = 1$ as in the original paper. Given an adversary $\mathcal{A}$ for the biased m-eTCR game, we construct an adversary $\mathcal{A}'$ for the average-case search problem for a boolean function $f : X_1 \times X_2 \rightarrow \{0, 1\}$:

- We sample a random $y \in Y'$
- We construct a random function H that returns y over $f^{-1}(1)$ and a random value in $Y \setminus \{y\}$ otherwise.
- When $\mathcal{A}$ makes its query to Box we sample x_1 so that $H(x_1, x_2) \in Y'$ and reprogram the oracle $(x_1, x_2) \mapsto y$.
- return $\mathcal{A}$'s output.

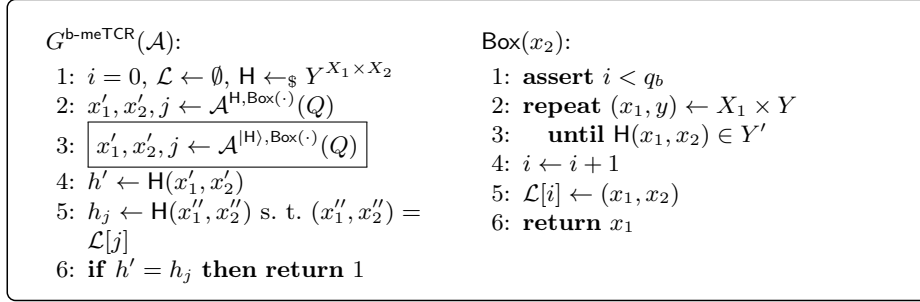


Fig. 10. Security game for the biased Multi-target eTCR problem. X_1, X_2, Y', Y are finite sets with $Y' \subset Y$. The adversary $\mathcal{A}$ is allowed to make at most q_b queries to the Box oracle and at most q_h queries to the random oracle H . The boxed line indicates the modification for the quantum setting, where the adversary is allowed quantum queries to H .

The only difference from [44, Figure 2] reduction is the sampling of y in the first step, but note that the output is still a solution to the average search problem. For a quantum adversary the advantage is unconditionally bounded by $\frac{8(q+1)^2}{|Y|}$ thanks to [44, Theorem 1], with q being the number of queries. A classical adversary instead has only black box access to a random function, thus it can only win with probability bounded by $q/|Y|$ with q the number of queries. For more details, we refer to [44, Appendix A]. In our case $q = q_b + q_h$. The multi-query case $q_b > 1$ can then be easily obtained, as confirmed in [19], and the winning probability is increased by a factor q_b .

Game G_1 : The second game is the original game from Figure 10. The only difference between the games is the reprogramming of the oracle. Thus, the difference $|\Pr[G_1 = 1] - \Pr[G_0 = 1]|$ can be bounded:

- for a quantum adversary by (8), using Corollary 1 as in the proof of [40, Theorem 2].
- for a classical adversary by

$$\frac{q_b(q_b + q_h)}{\gamma|X_2|}. \quad (11)$$

This because the adversary could have already queried a one of the q_b reprogrammed input (x_1, x_2) . $\square$

Thanks to the above tools, we can now prove the security of salt-PRISM in the (quantum) random oracle model.

Theorem 2. *Let $\mathcal{A}$ be an EUF-CMA adversary against salt-PRISM making at most N_{sign} signing queries and N_H (classical or quantum) queries to the random oracle $H : \text{Ell}_p \times \{0, 1\}^* \times \{0, 1\}^k \rightarrow \{0, 1\}^n$. Let $\gamma_{n,a} = \#\text{Primes}_a/2^n$. In the (quantum) random oracle model, there exists an adversary $\mathcal{B}$ against Problem 3*

for $N = N_{\text{sign}}$ such that:

$$\begin{aligned} \text{Adv}_{\text{salt-PRISM}}^{\text{EUF-CMA}}(\mathcal{A}) &\leq N_{\text{H}} \text{Adv}_{\text{Problem 3}}(\mathcal{B}) + \\ &\quad + \frac{(N_{\text{H}} + 1) \cdot N_{\text{sign}}}{2^n} + \frac{(N_{\text{sign}} + N_{\text{H}}) \cdot N_{\text{sign}}}{\gamma_{n,a} 2^{k-1}}. \end{aligned} \quad (12)$$

in the ROM, while

$$\begin{aligned} \text{Adv}_{\text{salt-PRISM}}^{\text{EUF-CMA}}(\mathcal{A}) &\leq (2N_{\text{H}} + 1)^2 \text{Adv}_{\text{Problem 3}}(\mathcal{B}) + \\ &\quad + \frac{N_{\text{sign}}(N_{\text{sign}} + N_{\text{H}} + 2)^2}{2^{n-3}} + 3N_{\text{sign}} \sqrt{\frac{N_{\text{H}} + N_{\text{sign}} + 1}{\gamma_{n,a} 2^k}}. \end{aligned} \quad (13)$$

in the QROM.

The proof will mimic the reduction from EUF-CMA to EUF-RMA from [40, Theorem 2], and then add another reduction step, similar to $\text{INV} \rightarrow \text{EUF-NMA}$ in [48, Theorem 4.1], that we use to force the adversary to output a forgery on a pre-sampled message.

Proof. Since we use an idealized model for the hash function, without loss of generality, we restrict the input space of H from $\text{Ell}_p \times \{0,1\}^* \times \{0,1\}^k$ to $X \times \{0,1\}^k$, where X is a finite set. We also write $\gamma_{n,a} = \#\text{Primes}_a / 2^n$.

Game $\mathbf{G}_0$: this is the standard EUF-CMA game for salt-PRISM.

Game $\mathbf{G}_1$: this game is identical to $\mathbf{G}_0$, except that if the adversary outputs a forgery (msg^*, σ^*) such that there exists a signing query (msg_i, σ_i) with $\text{H}_{\text{prime}}(E_{\text{vk}}, \text{msg}^*, \text{salt}^*) = \text{H}_{\text{prime}}(E_{\text{vk}}, \text{msg}_i, \text{salt}_i)$, then the game returns 0. Any adversary distinguishing between $\mathbf{G}_0$ and $\mathbf{G}_1$ can be used to obtain an adversary $\mathcal{A}'$ that breaks the biased multi-target eTRC security of the random oracle H (Figure 10) for $Y = 2^n$, $Y' = \text{Primes}_a$, $X_2 = X$ and $X_1 = \{0,1\}^k$. Thus, using Lemma 1 with $q_b = N_{\text{sign}}$ and $q_h = N_{\text{H}}$, we have can bound $|\Pr[\mathbf{G}_0(\mathcal{A}) = 1] - \Pr[\mathbf{G}_1(\mathcal{A}) = 1]|$.

Game $\mathbf{G}_2$: this game differs from $\mathbf{G}_1$ in the way the signing oracle works. In this game on the i -th signing queries, we repeatedly sample random $\text{salt}, h \in \{0,1\}^k \times \{0,1\}^n$ until $h \in \text{Primes}_a$, then we reprogram the random oracle at input $(\text{msg}, \text{salt})$. Note that here we are introducing a distribution over the salt string. Namely, for each input msg we sample salt uniformly random in the set of strings such that $H(\text{msg}, *) \in \text{Primes}_a$, of size about $\gamma_{n,a} 2^k$. If the entry was already reprogrammed we abort. Let $\varepsilon = |\Pr[\mathbf{G}_1(\mathcal{A}) = 1] - \Pr[\mathbf{G}_2(\mathcal{A}) = 1]|$. Then ε is as follows:

- Classically, ε is bounded by the probability that in $\mathbf{G}_1$ the signing oracle needs to reprogram an already reprogrammed entry. Since there are at most $N_{\text{sign}} + N_{\text{H}}$ reprogrammed entries and N_{sign} reprogramming attempts over a set of expected size $\gamma_{n,a} 2^k$,

$$\varepsilon \leq \frac{(N_{\text{sign}} + N_{\text{H}}) \cdot N_{\text{sign}}}{\gamma_{n,a} 2^k}. \quad (14)$$

- In the quantum setting, we can bound ε using [Corollary 1](#). Following the same reasoning as in [\[40, Theorem 2\]](#) we can easily write a reduction $\mathcal{R}$ handling the random oracle, keeping track of hash queries and relying signing queries to the REPROGRAM_b oracle from REPRO_b . If $\mathcal{R}$ plays REPRO_0 it simulates $\mathbf{G}_1$, while if it plays REPRO_1 it simulates $\mathbf{G}_2$. By [Corollary 1](#), we have that

$$\varepsilon \leq \frac{3N_H}{2} \cdot \sqrt{\frac{N_H + N_{\text{sign}} + 1}{\gamma_{n,a} 2^k}}. \quad (15)$$

Game $\mathbf{G}_3$: in this game we simulate the signing oracle using the input list from [Problem 3](#), i.e. a list of $N = N_{\text{sign}}$ isogenies $\{\phi_i : E \rightarrow E_i\}_{i=1}^N$ of degree $q_i(2^a - q_i)$ for q_i uniformly random in Primes_a and ϕ_i uniformly random among the isogenies of degree $q_i(2^a - q_i)$. For this, on the i -th signing query we substitute the output $h \in \text{Primes}_a$ from $\mathbf{G}_2$ with q_i from the input list, and return the corresponding isogeny ϕ_i as signature. Since both h and q_i are uniformly random in Primes_a , the adversary cannot distinguish between $\mathbf{G}_2$ and $\mathbf{G}_3$.

In the last step of our reduction, we consider the target prime $\bar{q}$, uniformly random in Primes_a , obtained as input from [Problem 3](#). We consider an algorithm $\mathcal{B}$ that interacts with $\mathcal{A}$ playing $\mathbf{G}_3$ and controls the random oracle $\mathbf{H}$. The goal of $\mathcal{B}$ is to encode $\bar{q}$ in the random oracle, so that with non-negligible probability, if the adversary outputs a forgery $(\text{msg}^*, \text{salt}^*, \sigma^*)$, then $\bar{q} = \mathbf{H}_{\text{prime}}(E_{\text{vk}}, \text{msg}^*, \text{salt}^*)$.

For this, we consider two different cases:

- Classically, consider the random oracle $\mathbf{H}$, we sample a random index $j^* \in [1, N_H]$. If the j^* -th random oracle query returns $h \in \text{Primes}_a$ we reprogram it to $\bar{q}$, otherwise we do nothing. The original and reprogrammed random oracles are indistinguishable, since both h and $\bar{q}$ are uniformly random in Primes_a . We assume that, without loss of generality, the adversary always returns a forgery $(\text{msg}^*, \text{salt}^*, \sigma^*)$ such that $\mathbf{H}_{\text{prime}}(E_{\text{vk}}, \text{msg}^*, \text{salt}^*)$ was queried to the random oracle and is in Primes_a (otherwise the forgery would be invalid). Thus, the probability that $\bar{q} = \mathbf{H}_{\text{prime}}(E_{\text{vk}}, \text{msg}^*, \text{salt}^*)$ is at least $1/N_H$, since the adversary makes at most N_H queries to the random oracle and j^* is uniformly random.

We conclude that

$$\text{Adv}_{\text{Problem 3}}(\mathcal{B}) \geq \frac{\Pr[\mathbf{G}_3(\mathcal{A}) = 1]}{N_H}. \quad (16)$$

- In the quantum setting, we can use the *Measure-and-Reprogram Technique* from [\[32, Theorem 2\]](#), as done in [\[48, Theorem 4.1\]](#). The only difference is that $\mathcal{B}$ needs to reprogram the random oracle to a random value in Primes_a , instead of $\{0, 1\}^n$. Let $\mathcal{B}$ act as the two stage simulator from [\[32, Theorem 2\]](#). In the first stage, $\mathcal{B}$ observes one of the random oracle queries $\mathcal{A}$ makes in the game $\mathbf{G}_3$, say $(\text{msg}', \text{salt}')$. If $\mathbf{H}(\text{msg}', \text{salt}') \in \text{Primes}_a$, we set $\theta := \bar{q}$,

otherwise we set θ uniformly random in $\{0, 1\}^n$. Then, $\mathcal{B}$ reprograms the random oracle as $H'(\text{msg}', \text{salt}') \mapsto \theta$. Note, by a straightforward application of conditional probability, θ is still uniformly random in Primes_a . In the second stage, $\mathcal{B}$ runs $\mathcal{A}$ again with H' , and obtains a forgery $(\text{msg}^*, \text{salt}^*, \sigma^*)$. After running $\mathcal{B}$, we output σ^* as a solution to [Problem 3](#) with target degree $\bar{q}$ if $\theta \in \text{Primes}_a$ and $\bar{q} = H'(\text{msg}^*, \text{salt}^*)$. From [\[32, Theorem 2\]](#), a reduction with a security loss of $(2N_H + 1)^2$ holds, so:

$$\text{Adv}_{\text{Problem 3}}(\mathcal{B}) \geq \frac{\Pr[\mathbf{G}_3(\mathcal{A}) = 1]}{(2N_H + 1)^2}. \quad (17)$$

As in the classical case, we can assume without loss of generality that the adversary always returns a forgery $(\text{msg}^*, \text{salt}^*, \sigma^*)$ such that $H_{\text{prime}}(E_{\text{vk}}, \text{msg}^*, \text{salt}^*)$ was queried to the random oracle and is in Primes_a (otherwise the forgery would be invalid).

We can now conclude the proof by combining all the bounds from the games. For the ROM case we use (9) from [Lemma 1](#), (14) and (16), while for the QROM case we use (10) from [Lemma 1](#), (15) and (17). $\square$

Remark 11. Note that our proof never uses the structure of the set Primes_a , only the ratio $\gamma_{n,a} = \#\text{Primes}_a/2^n$. It also does not use any property of the isogeny construction, only that the list of isogenies from [Problem 3](#) are valid signatures on random valid hash digests and that the challenge prime degree is uniformly random in the set of valid digests. Thus, the same proof can be applied to reduce EUF-CMA to EUF-RMA security with random challenge message for any a hash-and-sign signature scheme hashing to a subset of valid digests, in particular it can be straightforwardly applied to a salted version of the strongly unforgeable variant presented in [Section 5](#).

Parameters. We can use (12) from [Theorem 2](#) to give parameter choices for salt-PRISM. We only consider the case of classical adversaries, since the quadratic loss in the QROM seems to the best of our knowledge an artifact of the proof technique, also shared by other post-quantum hash-and-sign schemes [\[48\]](#). We set k , n and a such that the ratio between the winning probability of and the cost of the attack is negligible, i.e., $\leq 2^{-\lambda}$. To this end, we lower bound the cost of the attack as $N_{\text{sign}} + N_H$. From (12) we get the following constraints:

$$\frac{\text{Adv}_{\text{salt-PRISM}}^{\text{EUF-CMA}}(\mathcal{A})}{N_{\text{sign}} + N_H} \leq \text{Adv}_{\text{Problem 3}}(\mathcal{B}) + \frac{N_{\text{sign}}}{2^n} + \frac{N_{\text{sign}}}{\gamma_{n,a} 2^{k-1}} \leq 2^{-\lambda}. \quad (18)$$

Thus we can set $n \geq \lambda + \log_2(N_{\text{sign}})$ and $k \geq \lambda + \log_2(N_{\text{sign}}) - \log_2(\gamma_{n,a})$. As from NIST recommendations [\[55\]](#) we can set $N_{\text{sign}} = 2^{64}$, thus $n = \lambda + 64$ for any security level λ . We see that no constraint is imposed on a , but a larger a increases the size of the prime set Primes_a and $\gamma_{n,a}$, reducing the number repeated hashing attempts to find a valid salt, thus we set $a = n$.

Remark 12. Even if we are considering a relatively smaller value of a , the straightforward generalization to salt-PRISM of the attack in [53, Theorem 2] with complexity $O(2^{6a/7})$ does not impact the security salt-PRISM. In fact, $6a/7 \geq \lambda$ for λ up to 256, i.e., the highest NIST V security level. Also, note we are ignoring polynomial factors in the attack complexity.

Remark 13. In [6], the authors analyse the BUFF security of several signature schemes submitted to the current standardization process of additional signatures by NIST. In particular, they show that SQIsign only achieves $\lambda/2$ -bit security against message-bound signature (MBS)¹³ attacks. We remark that their attack can be similarly applied to salt-PRISM, and thus only achieves $a/2$ -bit security against MBS attacks.

For k we make the conservative choice of $k = 2\lambda$, which for little cost provides additional security margin and resistance against fault injections and side-channel attacks. Table 2 summarizes our parameter choices for salt-PRISM and impact on public key and signature sizes.

Remark 14. By further restricting the number of signatures or increasing the expected number of hash evaluation it would be possible to decrease the parameter a even further.

7 Implementation and performance

In this section, we evaluate the performance of our schemes, highlighting the differences between the signature scheme and the identification protocol. We prioritize efficiency and focus on the most performative variant: salt-PRISM. We provide a C implementation of salt-PRISM, and compare it with other recent isogeny-based signatures. The repository with the source code can be found at

https://github.com/mariascrs/PRISM_v2.

7.1 Parameter choices

Following the discussion of Section 4.4, we can now give concrete parameter choices for our schemes. To protect against endomorphism ring computation we require $p \approx 2^{2\lambda}$. To use the IdealTolsogeny algorithm, we also require access to the 2^e -torsion with $2^e \approx p$. This also allows us to represent isogenies of degree up to $2^e \approx 2^{2\lambda}$, satisfying the security requirements against isogeny computation and hash collisions for both salt-PRISM and PRISM-id.

We thus choose primes of the form $p = f2^e - 1$, with f a small cofactor. Such primes are commonly used in isogeny based schemes, for example in SQIsign NIST version 2 [1]. As such, we can follow their parameter choices and exploit the existing optimized implementations for those primes. We report these values for NIST security Levels I, III and V in Table 2, together with the respective public key and signature sizes discussed in the next section.

¹³ A modern reformulation of the non-repudiation property.

7.2 Sizes

The public key for PRISM-sig, salt-PRISM and PRISM-id is a curve E_{vk} . Since we are working in the Montgomery model, we can represent it with a single scalar in $\mathbb{F}_{p^2}$, with a cost of 4λ bits. A deterministic basis of the 2^a -torsion on E_{vk} can be included in the public key, to optimize verification performance, or computed on demand, to optimize compactness. This situation is completely analogous to SQIsign NIST version 2. As such, we follow their optimized approach of giving hints for the generation of the deterministic basis, with a cost of 2 bytes (see [9, Sec. 7]).

Signature Sizes. An isogeny representation consists of a curve E_{sig} and two points (P_{sig}, Q_{sig}) . We proceed as follows: first, we send both coordinates of P_{sig} . Since $\log p = 2\lambda$ and we are working in $\mathbb{F}_{p^2}$, we can represent these coordinates with 8λ bits. Since we work with Montgomery curves, from P_{sig} we can recover the curve E_{sig} at the cost of a single field inversion. We can then represent Q_{sig} by its x -coordinate plus a bit to indicate the sign of the y -coordinate, which we can recover by computing only a square root. Ignoring the sign bit, this requires a total of 12λ bits to represent the isogeny.

An alternative approach is to encode both P_{sig} and Q_{sig} by their coefficients with respect to a deterministic basis of the 2^a -torsion on E_{sig} . This is, for instance, the method adopted by SQIsign2D-West [9]. This approach requires 4λ bits for the Montgomery coefficient of E_{sig} and $4a$ bits for the points. Encoding the points in this way requires the prover to compute a deterministic basis for the 2^a -torsion on E_{sig} . This is more costly than computing a single square root, but hints can be provided as in the case of the public key.

A third approach is to send only three of the four coefficients above and recover the last one using pairings with the method discussed in [24, §6.1]. The signature would now have size $4\lambda + 3a$. However, this extra compactness comes at the additional cost of two pairing computations on top of the aforementioned deterministic basis.

For PRISM-sig we have $a \approx 2\lambda$, so the second approach comes with a (slight) loss in efficiency but no gain in size. The first approach seems better suited. On the other hand, for salt-PRISM a is smaller than 2λ , and the second approach makes the signature significantly shorter with a small increase in verification cost. We thus opt for this option. Notice that in this case the salt (of 2λ bits) must be added to the signature. Therefore, salt-PRISM signatures have size $6\lambda + 4a$.

Communication Costs for PRISM-id. The situation for PRISM-id is similar to salt-PRISM. From the discussion in Remark 9 we only need $a \approx \lambda + 2 \log \lambda$. This makes point compression more convenient. On top of the 4λ bits needed for the curve, it is enough to send four (or three, at the cost of pairing computations) coefficients of a bits each, for a total cost of $4\lambda + 4a \approx 8\lambda + 8 \log \lambda$ (resp. $\approx 7\lambda + 6 \log \lambda$). As above, the data reported in Table 2 refers to the efficiency-oriented version using four point coefficients.

Notice that the first interaction from Vrf only requires $a - 1$ bits to represent the prime $q \in \text{Primes}_a$; the total communication costs using point compression is then $4\lambda + 4a$.

Table 2. Public-key size and signature size in bytes, respectively communication cost, for the signature schemes **PRISM-sig** and **salt-PRISM**, respectively identification protocol **PRISM-id**.

	NIST I	NIST III	NIST V
Prime	$5 \cdot 2^{248} - 1$	$65 \cdot 2^{376} - 1$	$27 \cdot 2^{500} - 1$
Public-key size	66	98	130
PRISM-id a	144	210	274
PRISM-sig a	248	376	500
salt-PRISM a	192	256	320
PRISM-id com. cost	160	224	288
PRISM-sig sig. size	189	288	388
salt-PRISM sig. size	191	272	350

7.3 Comparison with other isogeny-based signatures

We now briefly compare the signature sizes of **salt-PRISM** and **PRISM-sig** with other isogeny-based signatures in [Table 3](#). For each scheme, we report the signature size of the main variant, ignoring logarithmic factors in terms of the security parameter λ . A more compact version of **SQIsign2D-East** presented in [\[54\]](#), **CompactSQIsign2D-East**, only requires 10λ bits. However, it comes with a significant computational overhead and an increase in the number of two-dimensional isogeny computations (up to about 770 for Level I). A trade-off similar to the one discussed above could bring **SQIsign2D-West** signature sizes from 9λ bits to 8λ bits [\[9, Remark 26\]](#). However, as in our case, the authors prefer the faster version of the scheme. Finally, for **SQIPrime** we report the size of the signature in the two-dimensional case. In their paper [\[34\]](#), the authors also present a four-dimensional version, which achieves a signature size of 12λ , but seems less practical due to the need for four-dimensional isogenies for verification.

Protocol	PRISM-sig	salt-PRISM	SQIsign	SQIsign2D-East	SQIsign2D-West	SQIPrime
Sig. size (bits)	12λ	$6\lambda + 4a$	$\approx 11\lambda$	12λ	9λ	19λ

Table 3. Signature sizes for the signature scheme given in this work, **SQIsign**, and its most efficient variants.

7.4 Performance

In this section, we compare the performance of **salt-PRISM** and **PRISM-sig** with other isogeny-based schemes. In Table 4 we report the number of isogeny computations in different dimensions performed by our schemes **salt-PRISM** and **PRISM-sig**, and various other SQIsign variants.

With respect to this metric, we see that our schemes have the most efficient signing among all the analyzed variants, with **salt-PRISM** achieving the smallest number of isogenies.

Regarding verification, for Level I **salt-PRISM** requires $192 = (\lambda + 64)$ 2-dimensional isogenies against the $126 \approx \lambda$ of SQIsign2D-West. However, SQIsign2D-West additionally requires $248 \approx 2\lambda$ one-dimensional isogenies. With a very rough estimate of one two-dimensional isogeny being as expensive as 4 one-dimensional isogenies (as we work with 4 theta coordinates in dimension 2) the cost of SQIsign2D-West can be estimated to be about $190 \approx 3\lambda/2$ 2-dimensional isogenies. This is slightly less than **salt-PRISM** at Level I, and slightly more at Level V. We thus expect **salt-PRISM** to perform faster than SQIsign2D-West in signing, and very close in verification.

To validate this theoretical analysis, we implemented **salt-PRISM** within the code base of the SQIsign NIST round 2 submission [1] using the Qlapoti ideal-to-isogeny algorithm [17]. The results are reported in Table 5.

Our signing procedure is between 1.4 and 1.6 times faster than SQIsign. This is already a significant improvement, but not as good as one may expect from Table 4. This can be explained by two factors: first, while being the overall bottleneck, isogeny computations are not the only operations involved in signing. The amount of quaternion operations involved in **salt-PRISM** is similar to SQIsign, thus bringing down the overall ratio. Second, we implemented **salt-PRISM** directly on the codebase of SQIsign. As a result, the underlying functions are specifically optimized to work with SQIsign parameters. A more thorough analysis of the distribution of costs and potential algorithmic optimizations is left as future work.

More interestingly, the verification time goes from 1.2 times slower to 1.01 times faster among the different security levels. This is the major improvement of **salt-PRISM** compared to **PRISM-sig**, and makes **salt-PRISM** faster than SQIsign in both signing and verification at Level V. Once again the ratio expected from a pure isogeny count would suggest that a slight optimization of the verification time is possible. The same considerations as above apply.

8 Conclusion

We present a novel two-round identification protocol and a hash-and-sign digital signature scheme, whose security reduces to the hardness of computing large prime degree isogenies from a curve with unknown endomorphism ring. The response and the signature consist of an efficient higher-dimensional representation of such isogeny with domain the verification key E_{vk} .

Table 4. Number of isogenies computed of each degree for NIST-I parameters. The numbers given in parentheses indicate that they may vary slightly depending on the case.

		Type of isogeny				
Protocol		2	3	5	(2, 2)	(2, 2, 2, 2)
salt-PRISM	KeyGen	-	-	-	496	-
	Sign	-	-	-	384	-
	Verify	-	-	-	192	-
PRISM-sig	KeyGen	-	-	-	496	-
	Sign	-	-	-	496	-
	Verify	-	-	-	248	-
SQIsignHD	KeyGen	378	234	-	-	-
	Sign	252	312	-	-	-
	Verify	-	78	-	-	142
SQIsign2D-West	KeyGen	-	-	-	496	-
	Sign	(248)	-	-	992	-
	Verify	(248)	-	-	(126)	-
SQIsign2D-West (Heuristic)	KeyGen	-	-	-	496	-
	Sign	(122)	-	-	624	-
	Verify	(122)	-	-	(126)	-
SQIsign2D-East	KeyGen	-	-	-	253	-
	Sign	127	(2)	(1)	641	-
	Verify	127	(2)	(1)	129	-

Table 5. Run time comparison in millions of clock cycles between salt-PRISM and SQIsign at NIST security levels I, III, and V. Averaged over 10,000 runs on an Intel Core Ultra 7 165H.

		Level I	Level III	Level V
PRISM (this work)	KeyGen	31.2	123.4	166.9
	Sign	50.3	172.1	260.0
	Verify	9.4	23.0	42.2
SQIsign v2 & Qlapoti	KeyGen	30.7	120.0	164.9
	Sign	76.1	280.4	407.1
	Verify	7.7	21.6	42.7

We note that the determining factor for secure parameters is actually the size of the challenge space and not the hardness of the underlying isogeny problem. In particular, for the signature scheme, we need to increase the set Primes_a to protect against collision attacks on the hash function. To mitigate this, we carefully measure the cost of computing the hash function H_{prime} , which involves repetitions and primality testing, to correctly assess the hardness of collision finding.

We evaluate the performance of the presented schemes, showing that they compare favorably to the original SQIsign and many of its variants. Furthermore, since our construction has the additional advantage of having a much simpler signature procedure, we hope future work will prove its usefulness as a building block for more advanced constructions.

References

1. Marius A. Aardal, Gora Adj, Diego F. Aranha, Andrea Basso, Isaac Andrés Canales Martínez, Jorge Chávez-Saab, Maria Corte-Real Santos, Pierrick Dartois, Luca De Feo, Max Duparc, Jonathan Komada Eriksen, Tako Boris Fouotsa, Décio Luiz Gazzoni Filho, Basil Hess, David Kohel, Antonin Leroux, Patrick Longa, Luciano Maino, Michael Meyer, Kohei Nakagawa, Hiroshi Onuki, Lorenz Panny, Sikhar Patranabis, Christophe Petit, Giacomo Pope, Krijn Reijnders, Damien Robert, Francisco Rodríguez-Henríquez, Sina Schaeffler, and Benjamin Wesolowski. SQIsign specification. <https://sqisign.org/spec/sqisign-20250205.pdf>, 2025. Accessed: 2025-12-08.
2. Marius A. Aardal, Andrea Basso, Luca De Feo, Sikhar Patranabis, and Benjamin Wesolowski. A complete security proof of SQIsign. In Yael Tauman Kalai and Seny F. Kamara, editors, *Advances in Cryptology – CRYPTO 2025, Part VI*, volume 16005 of *Lecture Notes in Computer Science*, pages 190–222, Santa Barbara, CA, USA, August 17–21, 2025. Springer, Cham, Switzerland.
3. Michel Abdalla, Jee Hea An, Mihir Bellare, and Chanathip Namprempre. From identification to signatures via the Fiat-Shamir transform: Minimizing assumptions for security and forward-security. In Lars R. Knudsen, editor, *Advances in Cryptology – EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 418–433, Amsterdam, The Netherlands, April 28 – May 2, 2002. Springer Berlin Heidelberg, Germany.
4. Jee Hea An, Yevgeniy Dodis, and Tal Rabin. On the security of joint signature and encryption. In Lars R. Knudsen, editor, *Advances in Cryptology – EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 83–107, Amsterdam, The Netherlands, April 28 – May 2, 2002. Springer Berlin Heidelberg, Germany.
5. Marcin Andrychowicz, Stefan Dziembowski, Daniel Malinowski, and Lukasz Mazurek. On the malleability of Bitcoin transactions. In Michael Brenner, Nicolas Christin, Benjamin Johnson, and Kurt Rohloff, editors, *FC 2015 Workshops*, volume 8976 of *Lecture Notes in Computer Science*, pages 1–18, San Juan, Puerto Rico, January 30, 2015. Springer Berlin Heidelberg, Germany.
6. Thomas Aulbach, Samed Düzl , Michael Meyer, Patrick Struck, and Maximiliane Weish upl. Hash your keys before signing - BUFF security of the additional NIST

- PQC signatures. In Markku-Juhani Saarinen and Daniel Smith-Tone, editors, *Post-Quantum Cryptography - 15th International Workshop, PQCrypto 2024, Part II*, pages 301–335, Oxford, UK, June 12–14, 2024. Springer, Cham, Switzerland.
7. Andrea Basso, Giacomo Borin, Wouter Castryck, Maria Corte-Real Santos, Riccardo Invernizzi, Antonin Leroux, Luciano Maino, Frederik Vercauteren, and Benjamin Wesolowski. PRISM: Simple and Compact Identification and Signatures from Large Prime Degree Isogenies. In Tibor Jager and Jiaxin Pan, editors, *Public-Key Cryptography – PKC 2025*, pages 300–332, Cham, 2025. Springer Nature Switzerland.
 8. Andrea Basso, Giacomo Borin, Wouter Castryck, Maria Corte-Real Santos, Riccardo Invernizzi, Antonin Leroux, Luciano Maino, Frederik Vercauteren, and Benjamin Wesolowski. PRISM: Simple And Compact Identification and Signatures From Large Prime Degree Isogenies. Cryptology ePrint Archive, Paper 2025/135, 2025.
 9. 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, *Advances in Cryptology – ASIACRYPT 2024, Part III*, volume 15486 of *Lecture Notes in Computer Science*, pages 339–370, Kolkata, India, December 9–13, 2024. Springer, Singapore, Singapore.
 10. Carsten Baum, Ward Beullens, Shibam Mukherjee, Emmanuela Orsini, Sebastian Ramacher, Christian Rechberger, Lawrence Roy, and Peter Scholl. One tree to rule them all: Optimizing GGM trees and OWFs for post-quantum signatures. Cryptology ePrint Archive, Paper 2024/490, 2024.
 11. Daniel J Bernstein, Luca De Feo, Antonin Leroux, and Benjamin Smith. Faster computation of isogenies of large prime degree. *Open Book Series*, 4(1):39–55, 2020.
 12. Daniel J. Bernstein, Daira Hopwood, Andreas Hülsing, Tanja Lange, Ruben Niederhagen, Louiza Papachristodoulou, Michael Schneider, Peter Schwabe, and Zooko Wilcox-O’Hearn. SPHINCS: Practical stateless hash-based signatures. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology – EUROCRYPT 2015, Part I*, volume 9056 of *Lecture Notes in Computer Science*, pages 368–397, Sofia, Bulgaria, April 26–30, 2015. Springer Berlin Heidelberg, Germany.
 13. Ward Beullens, Thorsten Kleinjung, and Frederik Vercauteren. CSI-FiSh: Efficient isogeny based signatures through class group computations. In Steven D. Galbraith and Shiho Moriai, editors, *Advances in Cryptology – ASIACRYPT 2019, Part I*, volume 11921 of *Lecture Notes in Computer Science*, pages 227–247, Kobe, Japan, December 8–12, 2019. Springer, Cham, Switzerland.
 14. 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, *Progress in Cryptology - INDOCRYPT 2014: 15th International Conference in Cryptology in India*, volume 8885 of *Lecture Notes in Computer Science*, pages 428–442, New Delhi, India, December 14–17, 2014. Springer, Cham, Switzerland.
 15. Dan Boneh and Victor Shoup. A graduate course in applied cryptography. *Draft 0.5*, 2020.
 16. Giacomo Borin, Yi-Fu Lai, and Antonin Leroux. Erebor and durian: Full anonymous ring signatures from quaternions and isogenies. Cryptology ePrint Archive, Report 2024/1185, 2024.
 17. Giacomo Borin, Maria Corte-Real Santos, Jonathan Komada Eriksen, Riccardo Invernizzi, Marzio Mula, Sina Schaeffler, and Frederik Vercauteren. Qlapoti: Simple and efficient translation of quaternion ideals to isogenies. Cryptology ePrint Archive, Report 2025/1604, 2025.

18. Joppe W. Bos, Léo Ducas, Eike Kiltz, Tancrede Lepoint, Vadim Lyubashevsky, John M. Schanck, Peter Schwabe, Gregor Seiler, and Damien Stehlé. CRYSTALS - Kyber: A CCA-secure module-lattice-based KEM. In *2018 IEEE European Symposium on Security and Privacy*, pages 353–367, London, United Kingdom, April 24–26, 2018. IEEE Computer Society Press.
19. Joppe W. Bos, Andreas Hülsing, Joost Renes, and Christine van Vredendaal. Rapidly verifiable XMSS signatures. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2021(1):137–168, 2021.
20. Wouter Castryck and Thomas Decru. An efficient key recovery attack on SIDH. In Carmit Hazay and Martijn Stam, editors, *Advances in Cryptology – EUROCRYPT 2023, Part V*, volume 14008 of *Lecture Notes in Computer Science*, pages 423–447, Lyon, France, April 23–27, 2023. Springer, Cham, Switzerland.
21. Maria Corte-Real Santos, Jonathan Komada Eriksen, Michael Meyer, and Krijn Reijnders. Aprèssqi: Extra fast verification for sqisign using extension-field signing. In Marc Joye and Gregor Leander, editors, *Advances in Cryptology - EUROCRYPT 2024 - 43rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zurich, Switzerland, May 26-30, 2024, Proceedings, Part I*, volume 14651 of *Lecture Notes in Computer Science*, pages 63–93. Springer, 2024.
22. Jolijn Cottaar, Steven D. Galbraith, Luciano Maino, and Monika Trimoska. An analysis of a weakened version of PRISM. *Cryptology ePrint Archive*, Paper 2026/906, 2026.
23. Cas Cremers, Samed Düzlül, Rune Fiedler, Marc Fischlin, and Christian Janson. BUFFing signature schemes beyond unforgeability and the case of post-quantum signatures. In *2021 IEEE Symposium on Security and Privacy*, pages 1696–1714, San Francisco, CA, USA, May 24–27, 2021. IEEE Computer Society Press.
24. Pierrick Dartois, Antonin Leroux, Damien Robert, and Benjamin Wesolowski. SQISignHD: New dimensions in cryptography. In Marc Joye and Gregor Leander, editors, *Advances in Cryptology – EUROCRYPT 2024, Part I*, volume 14651 of *Lecture Notes in Computer Science*, pages 3–32, Zurich, Switzerland, May 26–30, 2024. Springer, Cham, Switzerland.
25. Pierrick Dartois, Luciano Maino, Giacomo Pope, and Damien Robert. An algorithmic approach to $(2, 2)$ -isogenies in the theta model and applications to isogeny-based cryptography. In Kai-Min Chung and Yu Sasaki, editors, *Advances in Cryptology – ASIACRYPT 2024, Part III*, volume 15486 of *Lecture Notes in Computer Science*, pages 304–338, Kolkata, India, December 9–13, 2024. Springer, Singapore, Singapore.
26. Luca De Feo. Mathematics of isogeny based cryptography. *arXiv preprint arXiv:1711.04062*, 2017.
27. 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, *Advances in Cryptology – ASIACRYPT 2020, Part I*, volume 12491 of *Lecture Notes in Computer Science*, pages 64–93, Daejeon, South Korea, December 7–11, 2020. Springer, Cham, Switzerland.
28. Luca De Feo, Antonin Leroux, Patrick Longa, and Benjamin Wesolowski. New algorithms for the Deuring correspondence - towards practical and secure SQISign signatures. In Carmit Hazay and Martijn Stam, editors, *Advances in Cryptology – EUROCRYPT 2023, Part V*, volume 14008 of *Lecture Notes in Computer Science*, pages 659–690, Lyon, France, April 23–27, 2023. Springer, Cham, Switzerland.

29. Christian Decker and Roger Wattenhofer. Bitcoin transaction malleability and MtGox. In Mirosław Kutylowski and Jaideep Vaidya, editors, *ESORICS 2014: 19th European Symposium on Research in Computer Security, Part II*, volume 8713 of *Lecture Notes in Computer Science*, pages 313–326, Wrocław, Poland, September 7–11, 2014. Springer, Cham, Switzerland.
30. Christina Delfs and Steven D. Galbraith. Computing isogenies between supersingular elliptic curves over $\mathbb{F}_p$. *Designs, Codes and Cryptography*, 78(2):425–440, 2016.
31. Max Deuring. Die Typen der Multiplikatorenringe elliptischer Funktionenkörper. In *Abhandlungen aus dem mathematischen Seminar der Universität Hamburg*, volume 14, pages 197–272. Springer Berlin/Heidelberg, 1941.
32. Jelle Don, Serge Fehr, and Christian Majenz. The measure-and-reprogram technique 2.0: Multi-round fiat-shamir and more. In Daniele Micciancio and Thomas Ristenpart, editors, *Advances in Cryptology – CRYPTO 2020, Part III*, volume 12172 of *Lecture Notes in Computer Science*, pages 602–631, Santa Barbara, CA, USA, August 17–21, 2020. Springer, Cham, Switzerland.
33. Léo Ducas, Eike Kiltz, Tancrede Lepoint, Vadim Lyubashevsky, Peter Schwabe, Gregor Seiler, and Damien Stehlé. CRYSTALS-Dilithium: A lattice-based digital signature scheme. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2018(1):238–268, 2018.
34. 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, *Advances in Cryptology – ASIACRYPT 2024, Part III*, volume 15486 of *Lecture Notes in Computer Science*, pages 396–429, Kolkata, India, December 9–13, 2024. Springer, Singapore, Singapore.
35. Noam D. Elkies. Elliptic and modular curves over finite fields and related computational issues. In *Computational perspectives on number theory (Chicago, IL, 1995)*, volume 7 of *AMS/IP Studies in Advanced Mathematics*, pages 21–76. American Mathematical Society, 1998.
36. Jonathan Komada Eriksen, Lorenz Panny, Jana Sotáková, and Mattia Veroni. Deuring for the people: supersingular elliptic curves with prescribed endomorphism ring in general characteristic. In *LuCaNT: LMFDB, computation, and number theory*, volume 796 of *Contemporary Mathematics*, pages 339–373. American Mathematical Society, 2024.
37. Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In Andrew M. Odlyzko, editor, *Advances in Cryptology – CRYPTO’86*, volume 263 of *Lecture Notes in Computer Science*, pages 186–194, Santa Barbara, CA, USA, August 1987. Springer Berlin Heidelberg, Germany.
38. Pierre-Alain Fouque, Jeffrey Hoffstein, Paul Kirchner, Vadim Lyubashevsky, Thomas Pornin, Thomas Prest, Thomas Ricosset, Gregor Seiler, William Whyte, Zhenfei Zhang, et al. FALCON: Fast-Fourier lattice-based compact signatures over NTRU. *Submission to the NIST’s post-quantum cryptography standardization process*, 36(5):1–75, 2018.
39. Steven Galbraith. *Mathematics of public key cryptography*. Cambridge University Press, 2012.
40. Alex B. Grilo, Kathrin Hövelmanns, Andreas Hülsing, and Christian Majenz. Tight adaptive reprogramming in the QROM. In Mehdi Tibouchi and Huaxiong Wang, editors, *Advances in Cryptology – ASIACRYPT 2021, Part I*, volume 13090 of *Lecture Notes in Computer Science*, pages 637–667, Singapore, December 6–10, 2021. Springer, Cham, Switzerland.

41. Lov K. Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing*, STOC '96, pages 212–219, New York, NY, USA, 1996. Association for Computing Machinery.
42. Shai Halevi and Hugo Krawczyk. Strengthening digital signatures via randomized hashing. In Cynthia Dwork, editor, *Advances in Cryptology – CRYPTO 2006*, volume 4117 of *Lecture Notes in Computer Science*, pages 41–59, Santa Barbara, CA, USA, August 20–24, 2006. Springer Berlin Heidelberg, Germany.
43. G. H. Hardy and J. E. Littlewood. Some problems of "Partitio Numerorum"; III: On the expression of a number as a sum of primes. *Acta Mathematica*, 44(1):1–70, 1923.
44. Andreas Hülsing, Joost Rijneveld, and Fang Song. Mitigating multi-target attacks in hash-based signatures. In Chen-Mou Cheng, Kai-Min Chung, Giuseppe Persiano, and Bo-Yin Yang, editors, *PKC 2016: 19th International Conference on Theory and Practice of Public Key Cryptography, Part I*, volume 9614 of *Lecture Notes in Computer Science*, pages 387–416, Taipei, Taiwan, March 6–9, 2016. Springer Berlin Heidelberg, Germany.
45. Ernst Kani. The existence of curves of genus two with elliptic differentials. *Journal of Number Theory*, 64(1):130–161, 1997.
46. David Kohel. *Endomorphism rings of elliptic curves over finite fields*. PhD thesis, University of California, Berkeley, 1996.
47. David Kohel, Kristin Lauter, Christophe Petit, and Jean-Pierre Tignol. On the quaternion-isogeny path problem. *LMS Journal of Computation and Mathematics*, 17(A):418–432, 2014.
48. Haruhisa Kosuge and Keita Xagawa. Probabilistic hash-and-sign with retry in the quantum random oracle model. In Qiang Tang and Vanessa Teague, editors, *PKC 2024: 27th International Conference on Theory and Practice of Public Key Cryptography, Part I*, volume 14601 of *Lecture Notes in Computer Science*, pages 259–288, Sydney, NSW, Australia, April 15–17, 2024. Springer, Cham, Switzerland.
49. Johnson Lau and Pieter Wuille. BIP 146: Dealing with signature encoding malleability, 2016. Accessed: 2026-05-18.
50. Antonin Leroux. Verifiable random function from the deuring correspondence and higher dimensional isogenies. In Serge Fehr and Pierre-Alain Fouque, editors, *Advances in Cryptology – EUROCRYPT 2025, Part VII*, volume 15607 of *Lecture Notes in Computer Science*, pages 167–194, Madrid, Spain, May 4–8, 2025. Springer, Cham, Switzerland.
51. Luciano Maino, Chloe Martindale, Lorenz Panny, Giacomo Pope, and Benjamin Wesolowski. A direct key recovery attack on SIDH. In Carmit Hazay and Martijn Stam, editors, *Advances in Cryptology – EUROCRYPT 2023, Part V*, volume 14008 of *Lecture Notes in Computer Science*, pages 448–471, Lyon, France, April 23–27, 2023. Springer, Cham, Switzerland.
52. Gary L. Miller. Riemann's hypothesis and tests for primality. *Journal of Computer and System Sciences*, 13(3):300–317, 1976.
53. Kohei Nakagawa and Hiroshi Onuki. Attacks on PRISM-id via torsion over small extension fields. Cryptology ePrint Archive, Report 2025/1602, 2025.
54. 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, *Advances in Cryptology – ASIACRYPT 2024, Part III*, volume 15486 of *Lecture Notes in Computer Science*, pages 272–303, Kolkata, India, December 9–13, 2024. Springer, Singapore, Singapore.

55. NIST. Post-Quantum Cryptography Standardization, 2017. URL: <https://csrc.nist.gov/Projects/Post-Quantum-Cryptography>.
56. National Institute of Standards and Technology (NIST). Call for Additional Digital Signature Schemes for the Post-Quantum Cryptography Standardization Process, 2022.
57. Michael O. Rabin. Probabilistic algorithm for testing primality. *Journal of Number Theory*, 12(1):128–138, 1980.
58. Farzin Renan and Péter Kutas. SQIAsignHD: SQIsignHD adaptor signature. Cryptology ePrint Archive, Report 2024/561, 2024.
59. Damien Robert. Breaking SIDH in polynomial time. In Carmit Hazay and Martijn Stam, editors, *Advances in Cryptology – EUROCRYPT 2023, Part V*, volume 14008 of *Lecture Notes in Computer Science*, pages 472–503, Lyon, France, April 23–27, 2023. Springer, Cham, Switzerland.
60. Damien Robert. On the efficient representation of isogenies. Cryptology ePrint Archive, Paper 2024/1071, 2024. <https://eprint.iacr.org/2024/1071>.
61. Joseph H. Silverman. *The arithmetic of elliptic curves*, volume 106. Springer, 2009.
62. SQIsignHD code. <https://github.com/Pierrick-Dartois/SQISignHD-lib>.
63. Douglas R. Stinson. *Cryptography: theory and practice*. Chapman and Hall/CRC, 2005.
64. Paul C. van Oorschot and Michael J. Wiener. Parallel collision search with cryptanalytic applications. *Journal of Cryptology*, 12:1–28, 1999.
65. Jacques Vélú. Isogénies entre courbes elliptiques. *Comptes Rendus de l'Académie des Sciences de Paris*, 273:238–241, July 1971.
66. John Voight. *Quaternion algebras*, volume 288 of *Graduate Texts in Mathematics*. Springer, Cham, 2021.