

A Note on Constructing SIDH-PoK-based Signatures after Castryck-Decru Attack

Jesús-Javier Chi-Domínguez¹ 

Cryptography Research Center, Technology Innovation Institute, Abu Dhabi, UAE
jesus.dominguez@tii.ae

Abstract. In spite of the wave of devastating attacks on SIDH, started by Castryck-Decru (Eurocrypt 2023), there is still interest in constructing quantum secure SIDH Proofs of Knowledge (PoKs). For instance, SIDH PoKs for the Fixed Degree Relation, aim to prove the knowledge of a fixed degree d -isogeny ω between the elliptic curve E_0 and the public keys E_1, E_2 . In such cases, the public keys consist of only the elliptic curves (without image of auxiliary points), which suggests that the Castryck-Decru-like attack does not apply these scenarios.

In this paper we focus on the SIDH proof of knowledge of De Feo, Dobson, Galbraith, and Zobernig (Asiacrypt 2022); more precisely, we focus on their first 3-special soundness construction. In this work, we explicitly describe an optimized recoverable Σ -protocol based on their 3-special soundness SIDH-PoK. We also analyze the impact of building a signature scheme based on the optimized protocol and study the impact of moving to B-SIDH and G2SIDH setups, on the signature sizes.

Keywords: Isogeny-based cryptography · Proof-of-Knowledge · Σ -protocol · Signature scheme · Recoverable Σ -protocol

1 Introduction

In 2014, De Feo, Jao, and Plüt proposed the first post-quantum Diffie-Hellman protocol relying on the hardness of finding an isogeny between two supersingular curves, the SIDH protocol [36,23]. In addition, to key-exchange procedures; they also presented a Zero-Knowledge protocol based on the SIDH construction. Yoo, Azarderakhsh, Jalali, Jao, and Soukharev combined that Zero-Knowledge SIDH with the Fiat-Shamir transformation to get a signature scheme [46]. This was further improved in [34] to obtain shorter signatures.

Additionally, [21] presented an isogeny-based Proof of Knowledge (PoK) which was immune to known adaptive attacks [33,3,28,32] based on the same new hardness assumption. The main difference between [23,34] and [21] constructions is that they achieve 2-special and 3-special soundness, respectively.

Sadly, Castryck and Decru [12] recently presented a (heuristically) polynomial time SIDH key-recovery attack that breaks SIDH (and SIKE) in hours. The three vital ingredients for the applicability of the Castryck-Decru attack are

- The public and fixed isogeny degree;

- The image of the auxiliary torsion points under the secret isogeny; and
- The endomorphism ring of the isogeny domain curve.

The attack was subsequently improved by Maino and Martindale [41] and Robert [44]. Although these attacks [41,44] are theoretical; a public Magma code implementation of the attack in [12] was improved by Oudompheng and Pope [42]. It is worth mentioning that Castryck-Decru family of attacks apply to [36,23,46] but it does not extend to the construction from [21, §5.3] and the quaternion-based proposal of [34].

Contributions. In this work we focus on improving the communication efficiency (reducing proof-sizes) of the SIDH PoK with 3-special soundness. In particular, we reduce the proof-sizes of the protocol in [21, §5.3] and analyze the impact of using B-SIDH [17] and G2SIDH [38] in such a SIDH PoK protocol. As the main contribution, we optimize the Σ -protocol in [21, §5.3].

We provide detailed description of our optimizations, and show how to transform our optimized SIDH PoK into a signature scheme based on a non-interactive recoverable Σ -protocol. More importantly, we also discuss why the devastating Castryck-Decru attack (and its variants) do not threaten the security of SIDH PoK and our optimizations. We optimize the following variants of SIDH PoK:

- SIDH PoK from [21];
- B-SIDH variant of the SIDH PoK from [21]; and
- G2SIDH variant of the SIDH PoK from [21].

Related work. In 2019, De Feo and Galbraith proposed a signature scheme named SeaSign by combining the Commutative SIDH (CSIDH) [14] and Fiat-Shamir transformation with aborts [22]. This is followed by series of proposals with improved performance [26,6] or shorter sizes [29]. However, current proposals and implementations of these schemes use CSIDH-512, which may not achieve the desired quantum security of NIST Level 1 [8,43,7,15]. The current shortest isogeny-based signature scheme is SQIsign [24,25].

Outline. We present the necessary mathematical background related to SIDH in Section 2 and Section 2.1. We then provide an overview of the SIDH PoK protocol of [21] in Section 2.2. We discuss the (non-)impact of the recent attacks in Section 2.3. In Section 3 we provide details of the proposed optimizations (Section 3.1), followed by the optimized PoK protocol (Section 3.2) and the signature scheme (Section 3.3). We discuss the effect of replacing SIDH with B-SIDH in Section 3.4 and that of G2SIDH in Section 3.5 ¹.

2 Preliminaries

In this section, we introduce all mathematical tools required in the SIDH constructions from [36,23]. Let $p = 2^a 3^b - 1$ be a prime number satisfying $p \equiv$

¹ We highlight that we did not dig into the mathematical tools required for G2SIDH; we took it as a black box. However, we mention the main differences between SIDH and G2SIDH and take essential properties to describe how the recoverable Σ -protocol will impact.

$3 \pmod 4$ for some $a, b \in \mathbb{Z}^+$. Let \mathbb{F}_p be a prime field with p elements and \mathbb{F}_{p^2} a quadratic field extension of \mathbb{F}_p . We let E be a supersingular curve determined by Equation (1) and assume E has exactly $\#E(\mathbb{F}_{p^2}) = (p + 1)^2$ points over \mathbb{F}_{p^2} .

$$E: y^2 = x^3 + Ax^2 + x, \quad A \in \mathbb{F}_{p^2} \setminus \{\pm 2\}. \quad (1)$$

The point at infinity ∞ of E plays the role of the neutral element. We say $P \in E$ is an order- d point if d is the smallest positive integer such that

$$[d]P = \underbrace{P + \dots + P}_{d \text{ times}} = \infty,$$

and write $E[d]$ to denote the d -torsion subgroup $\{P \in E(\overline{\mathbb{F}_{p^2}}) \mid [d]P = \infty\}$. The j -invariant of the curve E is $\frac{256(A^2-3)^3}{A^2-4}$.

Isogenies From Kernel. We only consider separable isogenies. An isogeny $\phi: E \rightarrow E'$ over \mathbb{F}_{p^2} is a non-zero rational map fixing the point at infinity, $\phi(\infty) = \infty$. If such isogeny exists, we say E and E' are isogenous over \mathbb{F}_{p^2} , which happens if and only if $\#E(\mathbb{F}_{p^2}) = \#E'(\mathbb{F}_{p^2})$. The kernel $\ker \phi$ of ϕ is the subgroup $\{P \in E(\mathbb{F}_{p^2}) \mid \phi(P) = \infty\}$. We refer to ϕ as d -isogeny when $\#\ker \phi = d$ holds. The dual d -isogeny $\widehat{\phi}: E' \rightarrow E$ of ϕ is the isogeny satisfying

$$\widehat{\phi} \circ \phi: P \mapsto [d]P \quad \text{and} \quad \phi \circ \widehat{\phi}: P \mapsto [d]P.$$

We recall the following useful lemma from [21],

Lemma 1 (Lemma 2 [21](restated)). *Let $p = 2^a 3^b - 1$ be a prime number satisfying $p \equiv 3 \pmod 4$ for some $a, b \in \mathbb{Z}^+$. Let $\phi_A: E \rightarrow E_A$ be an isogeny of degree 2^a . Let $\phi_B: E \rightarrow E_B$ and $\phi_{AB}: E_A \rightarrow E_{AB}$ be isogenies of degree 3^b such that $\ker(\phi_{AB}) = \phi_A(\ker(\phi_B))$. Then there exists an isogeny $\phi_{BA}: E_B \rightarrow E_{AB}$ of degree 2^a .*

Moreover, as shown in the proof of Lemma 1 in [21], the isogeny ϕ_{AB} can be computed efficiently given, any generator of $\ker(\phi_A)$, ϕ_B , and E_B .

2.1 SIDH protocol

The core idea of [21, §5.3] relies on the SIDH-square construction. So, let us list the SIDH setup as follows:

- the public isogeny degrees $A = 2^a$ and $B = 3^b$;
- the quadratic field extension \mathbb{F}_{p^2} of \mathbb{F}_p along with $p = AB - 1$;
- the starting supersingular curve $E_0: y^2 = x^3 + 6x^2 + x^2$;

² We choose the same E_0 as in SIKE proposal [1], but it can be a different curve.

- the order-A basis $\{P_0, Q_0\}$ satisfying $\langle P_0, Q_0 \rangle = E_0[A]$; and
- the order-B basis $\{P'_0, Q'_0\}$ satisfying $\langle P'_0, Q'_0 \rangle = E_0[B]$.

The SIDH key generation is slightly different for each entity. Alice generates public keys according to order-B points, and her private keys determine secret A-isogenies. In contrast, Bob's public keys are concerning order-A points and his private keys to B-isogenies. We sketch as follows SIDH protocol.

Alice key generation.

1. Alice samples a random integer $\text{sk} \xleftarrow{\$} \llbracket 0 \dots A - 1 \rrbracket$ as her private key;
2. She then computes the A-isogeny $\phi: E_0 \rightarrow E_1$ with kernel generated by $K_\phi = P_0 + [\text{sk}]Q_0$; and
3. She sets as her public key $\text{pk} = (E_1, \phi(P'_0), \phi(Q'_0))$, and send it to Bob.

Bob key generation.

1. Bob samples a random integer $\text{sk}' \xleftarrow{\$} \llbracket 0 \dots B - 1 \rrbracket$ as his private key;
2. He then computes the B-isogeny $\psi: E_0 \rightarrow E_2$ with kernel generated by $K_\psi = P'_0 + [\text{sk}']Q'_0$; and
3. He sets as his public key $\text{pk}' = (E_2, \psi(P_0), \psi(Q_0))$, and send it to Alice.

Alice key derivation.

1. Alice computes the A-isogeny $\phi': E_2 \rightarrow E_3$ with kernel generated by $K_{\phi'} := \psi(K_\phi) = \psi(P_0) + [\text{sk}]\psi(Q_0)$; and
2. She finally sets as her secret shared the j-invariant $j(E_3)$ of E_3 .

Bob key derivation.

1. Bob computes the B-isogeny $\psi': E_1 \rightarrow E'_3$ with kernel generated by $K_{\psi'} := \phi(K_\psi) = \phi(P'_0) + [\text{sk}']\phi(Q'_0)$; and
2. He finally sets as his secret shared the j-invariant $j(E'_3)$ of E'_3 .

In the original SIDH construction from [36,23] and also in [1], the secret shared corresponds with the j-invariant of the curves E_3 and E'_3 . However, Leonardi showed that the ending curves E_3 and E'_3 are equal to each other [39]. We illustrate the diagram determined by the SIDH protocol in Figure 1.

Next, we summarize the constructions from [21] in Section 2.2. In particular, we only focus on the constructions based on Definition 1. The idea behind [21, §5.3] is to randomly generate SIDH-squares, as illustrated in Figure 1, to prove the knowledge of the secret isogeny.

Definition 1 (Fixed degree relation). *Given a public curve $\text{pk} = E_i$ generated by Alice or Bob without revealing any image of auxiliary points, we define the Fixed degree relation by Equation (2).*

$$\mathcal{R}_{\text{deg}} := \{(E_0, E_i, d, \omega) \mid \omega: E_0 \rightarrow E_i \text{ is a } d\text{-isogeny}\}. \quad (2)$$

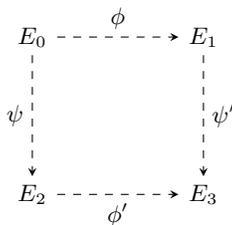


Fig. 1: Dashed arrows are secret and all curves are public. Horizontal and vertical arrows denote A-isogenies and B-isogenies, respectively. .

2.2 Overview of SIDH PoK

This section describes the construction from [21, §5.3]. The setup is the same as in Section 2.1. Given a public A-isogenous curve E_1 to E_0 . The prover (Peggy) wants to convince the verifier (Victor) that she knows the secret A-isogeny $\phi: E_0 \rightarrow E_1$, which is equivalent to knowing $\ker \phi = \langle K_\phi \rangle$, for the scope of this paper. In the following, we assume that C is statistically hiding and computationally binding commitment scheme.

Public and private keys. Here, the public key is $\text{pk} = E_1$, while $\text{sk} = \phi$ determines the private key.

Commitment. This block proceeds by constructing random SIDH-squares described in Figure 1 as follows.

- Peggy picks a random order-B kernel generator $K_\psi \in E_0[\mathbf{B}]$;
- She evaluates K_ψ under the secret isogeny ϕ to get $K_{\psi'} = \phi(K_\psi)$;
- She constructs an SIDH-square as in Figure 1 determined by
 - the B-isogeny $\psi: E_0 \rightarrow E_2$ with $\ker \psi = \langle K_\psi \rangle$,
 - the B-isogeny $\psi': E_1 \rightarrow E_3$ with $\ker \psi' = \langle K_{\psi'} \rangle$, and
 - the A-isogeny $\phi': E_2 \rightarrow E_3$ with $\ker \phi' = \langle K_{\phi'} \rangle$ where $K_{\phi'} = \psi(K_\phi)$;
- She chooses a random basis $\{P_2, Q_2\}$ of $E_2[\mathbf{B}]$;
- She evaluates P_2 and Q_2 under the secret isogeny ϕ' to get $P_3 = \phi'(P_2)$ and $Q_3 = \phi'(Q_2)$;
- She looks for two integers $c, d \in \llbracket 0 \dots \mathbf{B} - 1 \rrbracket$ such that
 - The dual isogeny $\hat{\psi}: E_2 \rightarrow E_0$ of ψ has kernel generator $K_{\hat{\psi}} = [c]P_2 + [d]Q_2$, and
 - The dual isogeny $\hat{\psi}': E_3 \rightarrow E_1$ of ψ' has kernel generator $K_{\hat{\psi}'} = [c]P_3 + [d]Q_3$;
- She selects three random numbers r_R, r_L , and r from $\{0, 1\}^{2\lambda}$.
- Next, She commits $\text{com}_2 = (E_2, P_2, Q_2)$ and $\text{com}_3 = (E_3, P_3, Q_3)$ as
 - $\text{com}_L = \mathbf{C}(\text{com}_2 \parallel r_L)$,
 - $\text{com}_R = \mathbf{C}(\text{com}_3 \parallel r_R)$, and
 - $\text{com}' = \mathbf{C}((c, d) \parallel r)$;

- Finally, She sends the commitment message $\text{com} \leftarrow (\text{com}_L, \text{com}_R, \text{com}')$ to Victor.

Challenge. Victor picks a uniformly random challenge $\text{chall} \xleftarrow{\$} \{-1, 0, 1\}$, and send it to Peggy.

Response. Once Peggy receives the challenge chall , she performs the following:

- If $\text{chall} = 1$, she sets $K_{\phi'} \leftarrow [z]K_{\phi}$ where z was randomly sampled from $\llbracket 0, A-1 \rrbracket$ such that $(z, A) = 1$, and sends $\text{resp} \leftarrow (\text{com}_2, r_L, K_{\phi'}, \text{com}_3, r_R)$ to Victor.
- If $\text{chall} = 0$, she sends $\text{resp} \leftarrow (\text{com}_3, r_R, c, d, r)$ to Victor.
- If $\text{chall} = -1$, she sends $\text{resp} \leftarrow (\text{com}_2, r_L, c, d, r)$ to Victor.

Verification. Depending on the challenge, Victor does the following calculations to validate the commitment and response:

- $(\text{com}_L, \text{com}_R, \text{com}') \leftarrow \text{com}$
- If $\text{chall} = 1$,
 - He parses
 - * $(\text{com}_2, r_L, K_{\phi'}, \text{com}_3, r_R) \leftarrow \text{resp}$,
 - * $(E_2, P_2, Q_2) \leftarrow \text{com}_2$, and
 - * $(E_3, P_3, Q_3) \leftarrow \text{com}_3$;
 - He **rejects** if $\text{C}(\text{com}_2 \parallel r_L) \neq \text{com}_L$ or $\text{C}(\text{com}_3 \parallel r_R) \neq \text{com}_R$;
 - He **rejects** if $K_{\phi'} \notin E_2$ or $K_{\phi'}$ does not have order A ;
 - He computes the A -isogeny $\phi' : E_2 \rightarrow E'_3$ with kernel generator $K_{\phi'}$;
 - Finally, Victor **accepts** if and only if $E_3 = E'_3$, $P_3 = \phi'(P_2)$ and $Q_3 = \phi'(Q_2)$, otherwise **rejects**.
- If $\text{chall} = 0$,
 - He parses
 - * $(\text{com}_3, r_R, c, d, r) \leftarrow \text{resp}$, and
 - * $(E_3, P_3, Q_3) \leftarrow \text{com}_3$;
 - Victor **rejects** if $\text{C}((c, d) \parallel r) \neq \text{com}'$ or $\text{C}(\text{com}_3 \parallel r_R) \neq \text{com}_R$;
 - He computes $K_{\widehat{\psi}}$ as $[c]P_3 + [d]Q_3$;
 - He **rejects** if $K_{\widehat{\psi}}$ does not have order B ;
 - He computes the B -isogeny $\widehat{\psi}' : E_3 \rightarrow E'_1$ with kernel generator $K_{\widehat{\psi}'}$;
 - Finally, Victor **accepts** if and only if $E_1 = E'_1$, otherwise **rejects**.
- If $\text{chall} = -1$,
 - He parses
 - * $(\text{com}_2, r_L, c, d, r) \leftarrow \text{resp}$, and
 - * $(E_2, P_2, Q_2) \leftarrow \text{com}_2$;
 - Victor **rejects** if $\text{C}(\text{com}_2 \parallel r_L) \neq \text{com}_L$ or $\text{C}((c, d) \parallel r) \neq \text{com}'$;
 - He computes $K_{\widehat{\psi}}$ as $[c]P_2 + [d]Q_2$;
 - He **rejects** if $K_{\widehat{\psi}}$ does not have order B ;
 - He computes the B -isogeny $\widehat{\psi} : E_2 \rightarrow E'_0$ with kernel generator $K_{\widehat{\psi}}$;
 - Finally, Victor **accepts** if and only if $E_0 = E'_0$, otherwise **rejects**.

Remark 1. The calculations in the **Response** and **Verification** when $\text{chall} = 1$ correspond with the horizontal arrows of Figure 1. While $\text{chall} = 0$ and $\text{chall} = -1$ determine the right-vertical and left-vertical arrows, respectively.

2.3 Effect of Recent Attacks on SIDH-PoK

The current wave of attacks by Castryck-Decru [12], Maino-Martindale [41], and Robert [44] do not extend to the Σ -protocol from [21, §5.3], which is described above in Section 2.2. Given that the public keys do not include images of any auxiliary point, those attacks do not help to find the secret isogeny ϕ :

- If `chall` = 1. The kernel generator $K_{\phi'}$ of $\phi': E_2 \rightarrow E_3$ is revealed, along with the points P_2, Q_2 and their respectively image $P_3 = \phi'(P_2)$ and $Q_3 = \phi'(Q_2)$. Therefore, any key-recovery attack from [12,41,44] recovers a kernel generator for the A-isogeny ϕ' , which is already public.
- If `chall` = 0. The kernel generator $K_{\widehat{\psi}'}$ of the (expected) dual B-isogeny $\widehat{\psi}': E_3 \rightarrow E_1$ is public, along with the image points $P_3 = \phi'(P_2)$ and $Q_3 = \phi'(Q_2)$. Now, the curve E_2 and the points $P_2, Q_2 \in E_2$ are not revealed, and thus the points P_3 and Q_3 looks like random points. Furthermore, there are no image of auxiliary points under ϕ' (or its dual). So, the current Castryck-Decru family attacks do not help to find the secret A-isogeny ϕ' .
- If `chall` = -1. The kernel generator $K_{\widehat{\psi}}$ of the (expected) dual B-isogeny $\widehat{\psi}: E_2 \rightarrow E_0$ is public, along with two random points P_2 and Q_2 . Now, the curve E_3 and the random points $P_3, Q_3 \in E_2$ are not revealed. In fact, there are no image of auxiliary points under ϕ (or its dual). So, the current Castryck-Decru family attacks do not help to find the secret A-isogeny ϕ .

2.4 Computational Assumptions Underlying SIDH PoK

The zero-knowledge property of the SIDH PoK described in Section 2.2 essentially relies on the computational hardness of distinguishing between well-formed and altered instances (E_2, E_3, ϕ') , this was introduced by [23] as a new security assumption in the form of Decisional Supersingular Product Problem (DSPP) as presented in Definition 2 below. We illustrate this in Figure 2. The transcript of the protocol do not leak any information (or the protocol achieves special honest-verifier zero-knowledge property) as long as the cases from Figure 2a, Figure 2b, and Figure 2c do not simultaneously occur for any given fixed instance.

Definition 2 (Decisional Supersingular Product Problem (DSPP): Alice’s case). *Let E_0 be a Montgomery curve as in the SIDH setting (see Section 2.1). Given a A-isogeny $\phi: E_0 \rightarrow E_1$ with kernel $\langle K_\phi \rangle$, the Decisional Supersingular Product Problem (DSPP) asks to distinguish between the following two distributions:*

- (E_2, E_3, ϕ') is the bottom of a random SIDH-square as in Figure 1. That is, for a randomly chosen order-B kernel $\langle K_\psi \rangle$, we have E_2 is the codomain curve of the B-isogeny ψ with kernel $\langle K_\psi \rangle$, E_3 is the codomain curve of the B-isogeny ψ' with kernel $\langle \phi(K_\psi) \rangle$, and $\phi': E_2 \rightarrow E_3$ is the A-isogeny with kernel $\langle \psi(K_\phi) \rangle$.
- (E_2, E_3, ϕ') such that E_2 is a randomly chosen elliptic curve with same cardinality as E_0 , and $\phi': E_2 \rightarrow E_3$ is a random A-isogeny with cyclic kernel.

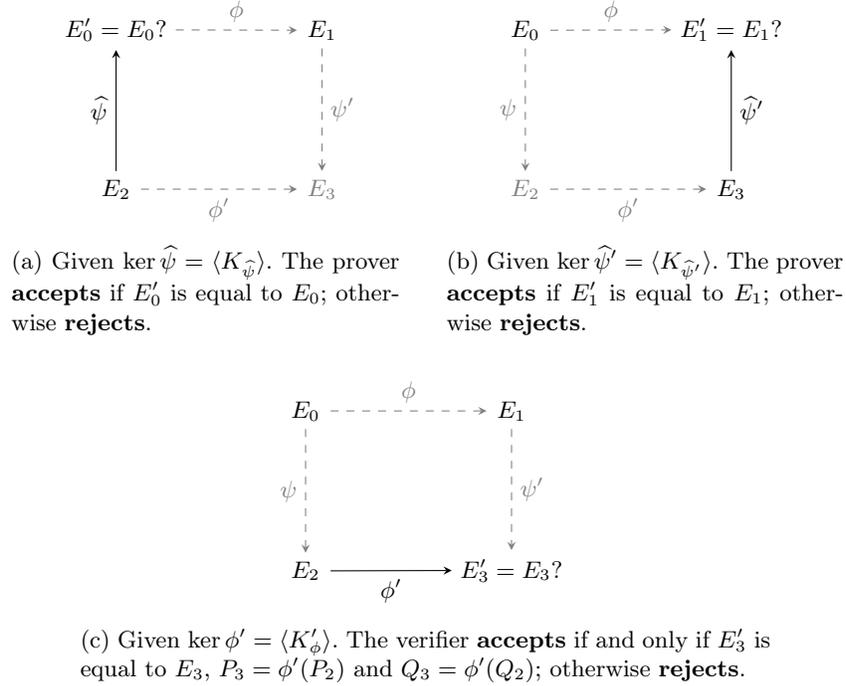


Fig. 2: Dashed arrows and curves labeled with gray ink are secret and unknown by the adversary and distinguisher.

The Σ -protocol described in Section 2.2 is 3-special soundness under the relation given by Definition 1. Furthermore, when repeated κ times, it becomes a Special Honest-Verifier Zero-Knowledge (SHVZK) PoK with soundness error $(2/3)^\kappa$, assuming the DSPP is computationally hard and the commitment scheme C is computationally binding and statistically hiding [21, Theorem 4].

3 Optimizing SIDH PoK

This section describes a way to optimize the Sigma construction described in Section 2.2 by applying the tricks from [2,18,34].

3.1 Reducing sizes according state-of-the-art tricks

Let $\lambda \in \{128, 192, 256\}$ a security parameter, and C be a statistically hiding and computationally binding commitment scheme with output length 2λ . The commitment $\text{com} = (\text{com}_L, \text{com}_R, \text{com}')$ has fixed bit-length equals 6λ . Recall

- $\text{com}_L = C(\text{com}_2 \parallel r_L)$ with $\text{com}_2 = (E_2, P_2, Q_2)$;
- $\text{com}_R = C(\text{com}_3 \parallel r_R)$ with $\text{com}_3 = (E_3, P_3, Q_3)$; and

– $\text{com}' = \mathbb{C}((c, d) \parallel r)$ where $K_\psi = [c]P_2 + [d]Q_2$ and $K_{\psi'} = [c]P_3 + [d]Q_3$ hold.

The response resp has a different size depending on if $\text{chall} = 1$ holds; let us analyze the cases below:

Vertical Case. This case corresponds to when $\text{chall} \neq 1$. The response includes $\log_2(p)$ bits that determines (c, d) . Notice, we can do it better by computing either $\Delta = (cd^{-1} \bmod \mathbb{B})$ or $\Delta = (dc^{-1} \bmod \mathbb{B})$ plus one bit $b \in \{0, 1\}$ to decide which point is multiplied by Δ : either $P_j + [\Delta]Q_j$ or $[\Delta]P_j + Q_j$ as kernel point generator for $j := 2, 3$. In other words, we suggest to replace (c, d) by (b, Δ) , and update the commit com' as $\mathcal{H}((b, \Delta) \parallel r)$. That trick reduces (c, d) of $\log_2(p)$ bits to (b, Δ) of $\frac{\log_2(p)}{2}$ bits. Now, let $\text{CanonicalBasis}_{\mathbb{B}}(E_j)$ denotes the procedure to find two order- \mathbb{B} points P and Q such that $\langle P, Q \rangle = E_j[\mathbb{B}]$, and set $j \in \{2, 3\}$. The commitment $\text{com}_j = (E_j, P_j, Q_j)$ has $10 \log_2(p)$ bits. The idea is to compute $P, Q \leftarrow \text{CanonicalBasis}_{\mathbb{B}}(E_j)$ and find integers $\alpha_{P_j}, \alpha_{Q_j}, \beta_{P_j}, \beta_{Q_j} \in \llbracket 0 \dots \mathbb{B} - 1 \rrbracket$ such that $P_j = [\alpha_{P_j}]P + [\beta_{P_j}]Q$ and $Q_j = [\alpha_{Q_j}]P + [\beta_{Q_j}]Q$. Therefore, replace the commitment $\text{com}_j = (E_j, P_j, Q_j)$ by $\text{com}_j = (E_j, (\alpha_{P_j}, \beta_{P_j}), (\alpha_{Q_j}, \beta_{Q_j}))$. That trick reduces the sizes from $10 \log_2(p)$ bits to about $4 \log_2(p)$ bits.

Horizontal Case. The response includes both com_2 and com_3 , along with the kernel order- \mathbb{A} point generator $K_{\phi'}$. Same trick as in the case $\text{chall} \neq 1$ allows to reduce the commitment size of $(\text{com}_2, \text{com}_3)$ from $20 \log_2(p)$ to $6 \log_2(p)$. In our optimized protocol we only send com_2 , since we can get P_3 and Q_3 from P_2 and Q_2 using ϕ' . Let $\text{CanonicalBasis}_{\mathbb{A}}(E_2)$ denotes the procedure to find two order- \mathbb{A} points P' and Q' such that $\langle P', Q' \rangle = E_2[\mathbb{A}]$. Analogously to the \mathbb{B} -torsion basis case, we can reduce $K_{\phi'}$ by finding two integers $\alpha, \beta \in \llbracket 0 \dots \mathbb{A} - 1 \rrbracket$ such that $K_{\phi'} = [\alpha]P' + [\beta]Q'$. Moreover, we suggest to represent $K_{\phi'}$ using $\frac{\log_2 p}{2}$ by computing either $\Delta' = (\alpha\beta^{-1} \bmod \mathbb{A})$ or $\Delta' = (\beta\alpha^{-1} \bmod \mathbb{A})$ plus one bit $b' \in \{0, 1\}$ to decide which point is multiplied by Δ' : either $P' + [\Delta']Q'$ or $[\Delta']P' + Q'$ as kernel point generator. Therefore in this case we send only $\frac{\log_2 p}{2}$ bits instead of $4 \log_2(p)$ bits.

3.2 Optimized Σ -protocol

We now show the above mentioned optimizations can be used to reduce the communication SIDH PoK. Let us assume Peggy wants to convince Victor that she knows the secret \mathbb{A} -isogeny $\phi: E_0 \rightarrow E_1$, which implies knowing $\ker \phi = \langle K_\phi \rangle$. We present the optimized Σ -protocol for the same in Figure 3 and the verification algorithm in Figure 4. The following lemmas, related to the knowledge-soundness and zero-knowledge properties of our optimized protocol. The correctness of the protocol follows from the construction of SIDH square in Figure 1.

Lemma 2 (Knowledge Soundness). *The optimized protocol in Figure 3 is knowledge sound with knowledge error $\frac{2}{3}$.*

Public and private keys. Here, the public key is $\text{pk} = E_1$, while $\text{sk} = \phi$ determines the private key.

Commitment. This block proceeds by constructing random SIDH-squares described in Figure 1 as follows.

- Peggy picks a random order-B kernel generator K_ψ in E_0 ;
- She evaluates K_ψ under the secret isogeny ϕ to get $K_{\psi'} = \phi(K_\psi)$;
- She constructs an SIDH-square as in Figure 1 determined by
 - the B-isogeny $\psi: E_0 \rightarrow E_2$ with $\ker \psi = \langle K_\psi \rangle$,
 - the B-isogeny $\psi': E_1 \rightarrow E_3$ with $\ker \psi' = \langle K_{\psi'} \rangle$, and
 - the A-isogeny $\phi': E_2 \rightarrow E_3$ with $\ker \phi' = \langle K_{\phi'} \rangle$ where $K_{\phi'} = \psi(K_\phi)$;
- She sets $K_{\phi'} \leftarrow [z]K_{\phi'}$ where z was randomly sampled from $\llbracket 0, A-1 \rrbracket$ such that $(z, A) = 1$;
- She chooses a random basis $\{P_2, Q_2\}$ of $E_2[\mathbf{B}]$;
- She evaluates P_2 and Q_2 under the secret isogeny ϕ' to get $P_3 = \phi'(P_2)$ and $Q_3 = \phi'(Q_2)$;
- She looks for two integers $c, d \in \llbracket 0 \dots \mathbf{B}-1 \rrbracket$ such that
 - The dual isogeny $\hat{\psi}: E_2 \rightarrow E_0$ of ψ has kernel generator $K_{\hat{\psi}} = [c]P_2 + [d]Q_2$,
 - The dual isogeny $\hat{\psi}': E_3 \rightarrow E_1$ of ψ' has kernel generator $K_{\hat{\psi}'} = [c]P_3 + [d]Q_3$;
- She selects three random numbers r_R, r_L , and r from $\{0, 1\}^{2\lambda}$;
- Next, She commits $\text{com}_2 = (E_2, P_2, Q_2)$ and $\text{com}_3 = (E_3, P_3, Q_3)$ as
 - $\text{com}_L = C(\text{com}_2 \parallel r_L)$,
 - $\text{com}_R = C(\text{com}_3 \parallel r_R)$, and

- $\text{com}' = C((b, \Delta) \parallel r)$: (b, Δ) are computed as in Section 3.1 vertical case;

- Finally, She sends the commitment message $\text{com} \leftarrow (\text{com}_L, \text{com}_R, \text{com}')$ to Victor.

Challenge. Victor picks a uniformly random challenge $\text{chall} \xleftarrow{\$} \{-1, 0, 1\}$, and send it to Peggy.

Response. Once Peggy receives the challenge chall , she performs the following:

- If $\text{chall} = 1$, she gets (b', Δ') , $(\alpha_{P_j}, \beta_{P_j})$, and $(\alpha_{Q_j}, \beta_{Q_j})$ for $j := 2, 3$ as in Section 3.1 horizontal case, and sets

$$\text{resp} \leftarrow (E_2, (\alpha_{P_2}, \beta_{P_2}), (\alpha_{Q_2}, \beta_{Q_2}), r_L, (b', \Delta'), E_3, r_R);$$

- If $\text{chall} = 0$, she obtains $(\alpha_{P_3}, \beta_{P_3})$ and $(\alpha_{Q_3}, \beta_{Q_3})$ as in Section 3.1 vertical case, and sets

$$\text{resp} \leftarrow (E_3, (\alpha_{P_3}, \beta_{P_3}), (\alpha_{Q_3}, \beta_{Q_3}), r_R, (b, \Delta), r);$$

- If $\text{chall} = -1$, she computes $(\alpha_{P_2}, \beta_{P_2})$ and $(\alpha_{Q_2}, \beta_{Q_2})$ as in Section 3.1 vertical case, and sets

$$\text{resp} \leftarrow (E_2, (\alpha_{P_2}, \beta_{P_2}), (\alpha_{Q_2}, \beta_{Q_2}), r_L, (b, \Delta), r);$$

Fig. 3: Optimized Σ -protocol

Verification. Depending on the challenge, Victor does the following calculations to validate the commitment and response:

- Victor parses the first received message as: $(\text{com}_L, \text{com}_R, \text{com}') \leftarrow \text{com}$
- If $\text{chall} = 1$,
 - He parses the response **resp** as $(E_2, (\alpha_{P_2}, \beta_{P_2}), (\alpha_{Q_2}, \beta_{Q_2}), r_L, (b', \Delta'), E_3, r_R)$.
 - He deterministically computes $(P, Q) \leftarrow \text{CanonicalBasis}_B(E_2)$, and $(P', Q') \leftarrow \text{CanonicalBasis}_A(E_2)$.
 - Victor also computes kernel generator $K_{\phi'}$ with the help of P', Q' and (b', Δ') as in Section 3.1 horizontal case.
 - He computes the A-isogeny $\phi' : E_2 \rightarrow E'_3$ with kernel generator $K_{\phi'}$.
 - He then computes, $\text{com}_2 = (E_2, P_2 = [\alpha_{P_2}]P + [\beta_{P_2}]Q, Q_2 = [\alpha_{Q_2}]P + [\beta_{Q_2}]Q)$ and $\text{com}_3 = (E_3, P_3 = \phi'(P_2), Q_3 = \phi'(Q_2))$.
 - He **rejects** if $\mathbb{C}(\text{com}_2 \parallel r_L) \neq \text{com}_L$ or $\mathbb{C}(\text{com}_3 \parallel r_R) \neq \text{com}_R$;
 - Finally, Victor **accepts** if and only if $E_3 = E'_3$, otherwise **rejects**.
- If $\text{chall} = 0$,
 - He parses the response **resp** as $(E_3, (\alpha_{P_3}, \beta_{P_3}), (\alpha_{Q_3}, \beta_{Q_3}), r_R, (b, \Delta), r)$.
 - He deterministically computes $(P, Q) \leftarrow \text{CanonicalBasis}_B(E_3)$,
 - Victor also computes kernel generator $K_{\hat{\psi}'}$ with the help of P, Q and (b, Δ) as in Section 3.1 vertical case.
 - He then computes, $\text{com}_3 = (E_3, P_3 = [\alpha_{P_3}]P + [\beta_{P_3}]Q, Q_3 = [\alpha_{Q_3}]P + [\beta_{Q_3}]Q)$
 - Victor **rejects** if $\mathbb{C}((b, \Delta) \parallel r) \neq \text{com}'$ or $\mathbb{C}(\text{com}_3 \parallel r_R) \neq \text{com}_R$;
 - He computes the B-isogeny $\hat{\psi}' : E_3 \rightarrow E'_1$ with kernel generator $K_{\hat{\psi}'}$;
 - Finally, Victor **accepts** if and only if $E_1 = E'_1$, otherwise **rejects**.
- If $\text{chall} = -1$,
 - He parses the response **resp** as $(E_2, (\alpha_{P_2}, \beta_{P_2}), (\alpha_{Q_2}, \beta_{Q_2}), r_L, (b, \Delta), r)$.
 - He deterministically computes $(P, Q) \leftarrow \text{CanonicalBasis}_B(E_2)$,
 - Victor also computes kernel generator $K_{\hat{\psi}}$ with the help of P, Q and (b, Δ) as in Section 3.1 vertical case.
 - He then computes, $\text{com}_2 = (E_2, P_2 = [\alpha_{P_2}]P + [\beta_{P_2}]Q, Q_2 = [\alpha_{Q_2}]P + [\beta_{Q_2}]Q)$
 - Victor **rejects** if $\mathbb{C}((b, \Delta) \parallel r) \neq \text{com}'$ or $\mathbb{C}(\text{com}_2 \parallel r_L) \neq \text{com}_L$;
 - He computes the B-isogeny $\hat{\psi}' : E_2 \rightarrow E'_0$ with kernel generator $K_{\hat{\psi}}$;
 - Finally, Victor **accepts** if and only if $E_0 = E'_0$, otherwise **rejects**.

Fig. 4: Verification algorithm for Optimized Σ -protocol

Lemma 3 (Special Honest Verifier Zero-Knowledge). *Let \mathcal{C} be a statistically hiding and computationally binding commitment scheme. Assuming the hardness of DSPP the optimized protocol in Figure 3 is computationally SHVZK in the random oracle model.*

The proofs of Lemma 2 and Lemma 3 are identical to the proof of [21, Theorem 2] with necessary changes required to compute the intermediate values from the responses of our optimized protocol. For completeness, we present these proofs in Section A of the appendix.

3.3 Signature Scheme Based On Optimized SIDH PoK

We now show the optimized Σ -protocol in Section 3.2 can be transformed into a signature scheme by using Fiat-Shamir transform [30].

Signature scheme using the strong Fiat-Shamir transform [30,4]. The main idea is to avoid the interaction between Peggy and Victor by allowing Peggy to generate the challenge as the hash of the statement and the commitment. In our case, Peggy would first generate κ commitments com_i and then obtains the challenge $(\text{chall}_1, \dots, \text{chall}_{\kappa-1}) = \mathcal{RO}(\text{pk}, m, \text{com}_0, \dots, \text{com}_{\kappa-1})$, where m is the message to be signed. We denote by \mathcal{RO} a random oracle that outputs strings in $\{-1, 0, 1\}^\kappa$. Each challenge chall_i determines the response values for com_i . This transformation is secure [45] in the Quantum Random Oracle Model.

Reducing sizes via recoverable Σ -protocol. Following the hints from [5, c.f. Remark 3], we transform the Σ -protocol into a recoverable Σ -protocol. That is, the signer can output $(\text{chall}, \text{resp})$ as signature instead of $(\text{com}, \text{resp})$. Given a signature $(\text{chall}, \text{resp})$, Victor then first recomputes com , and checks that $\text{chall} = \mathcal{H}(\text{pk} || m || \text{com})$ before verifying the transcript.

Let E_0 be a public parameter curve and (pk, sk) be generated by running the key generation algorithm described below. Also, let m be a message to be signed. In the following we assume that the commitment algorithm \mathcal{C} is instantiated with the help of random oracle \mathcal{H} . As required, the commitments generated this way are statistically hiding and computationally binding. In practice, we use cryptographic (collision-resistant) hash functions to generate the commitments, which are modelled as random oracle for security analysis. While converting the optimized Σ -protocol to signature scheme via Fiat-Shamir transform, we also use domain separation to ensure the random oracles used in different phases of the signing and verification are independent of each other.

Key Generation. Given the public curve E_0 , Peggy randomly samples an order- A kernel generator $K_\phi \in E_0[A]$, and computes the A -isogeny $\phi: E_0 \rightarrow E_1$ with kernel $\ker \phi = \langle K_\phi \rangle$. The public key is $\text{pk} = E_1$, while $\text{sk} = \phi$ is the private key.

Signing. Peggy proceeds as follows:

- She computes $\text{com} = (\text{com}_L, \text{com}_R, \text{com}')$ as given in the commitment phase of the optimized Σ -protocol in Figure 3.

- She calculates $\text{com}_{\mathcal{H}} \leftarrow \mathcal{H}(\text{pk} \parallel m \parallel \text{com})$ with $\text{com} = (\text{com}_L, \text{com}_R, \text{com}');$
- She picks as random challenge as $\text{chall} \leftarrow \text{PRNG}(\text{com}_{\mathcal{H}}) \in \{-1, 0, 1\};$
- She then creates the appropriate response resp' corresponding to chall as shown in the response phase of the optimized Σ -protocol in Figure 3.
- Based on the value of chall , she sets the response as follows:
 - If $\text{chall} = 1$: $\text{resp} \leftarrow (\text{com}', \text{resp}')$
 - If $\text{chall} = 0$: $\text{resp} \leftarrow (\text{com}_L, \text{resp}')$
 - If $\text{chall} = -1$: $\text{resp} \leftarrow (\text{com}_R, \text{resp}')$
- Finally, Peggy sends $\sigma \leftarrow (\text{chall}, \text{resp})$ to Victor.

Verifying. During the verification of the signature, Victor first computes $\text{com} = (\text{com}_L, \text{com}_R, \text{com}')$ from the received response resp . He then proceeds as in the verification algorithm in Figure 4. Finally before accepting he additionally checks if, $\text{PRNG}(\mathcal{H}(\text{pk} \parallel m \parallel \text{com})) = \text{chall}$. If all the checks pass, then Victor accepts.

The signature scheme described above is existentially unforgeable against the chosen message attacks. The proof of security follows the standard proof for constructing signatures via Fiat-Shamir transform for identification protocols.

Signature Sizes. Notice, if $\text{chall} = 1$ then the response resp in the above recoverable Σ -protocol has $\frac{12\lambda+13\log_2(p)}{2}$ bits; otherwise, it has $\frac{12\lambda+9\log_2(p)}{2}$ bits. Therefore, on average, the response resp has $\frac{36\lambda+31\log_2(p)}{6} \approx (6\lambda + 6\log_2(p))$ bits. As the last optimization, we suggest taking

$$(\text{chall}_0, \dots, \text{chall}_{\kappa-1}) \leftarrow \mathcal{RO}(\mathcal{H}'(\text{com}_{\mathcal{H},0}, \dots, \text{com}_{\mathcal{H},\kappa-1}))$$

as κ challenges for κ repetitions of the above recoverable Σ -protocol, where \mathcal{H}' is another hash function returning 2λ -bits and \mathcal{RO} is a random oracle that uniformly samples from $\{-1, 0, 1\}^\kappa$. After that, we get a signature

$$\sigma = (\mathcal{H}'(\text{com}_{\mathcal{H}',0}, \dots, \text{com}_{\mathcal{H},\kappa-1}), \text{resp}_0, \dots, \text{resp}_{\kappa-1})$$

of $(2\lambda + 6\kappa(\lambda + \log_2(p)))$ -bits. In Table 1 we follow [1,40] to estimate the sizes.

3.4 Over the quadratic twist

Following B-SIDH construction [17,19], we analyze the signature sizes using the quadratic twist curve. For instance, according to the parameter sets from [19], we can use primes of 256-bits (NIST Level 1), 384-bits (NIST Level 3), and 512-bits (NIST Level 5). The idea is to choose a prime number p with $A \mid (p+1)$ and $B \mid (p-1)$ being smooth integer numbers close to p . Thereafter, [21, Theorem 4] also holds if we repeat κ times the Σ -protocol described in [21, §5.3] and replace $A = 2^a$ and $B = 3^b$ with $A \mid (p+1)$ and $B \mid (p-1)$, respectively. It becomes an SHVZK PoK with soundness $(2/3)^\kappa$, assuming the DSPP is computationally hard.

This time, each integer coefficient given in the response resp has $\log_2(p)$ bits instead of $\frac{\log_2(p)}{2}$. Therefore, if $\text{chall} = 1$, we have a resp of $(6\lambda + 13\log_2(p))$ bits; otherwise, we have resp of $(6\lambda + 9\log_2(p))$ bits. Moreover, on average, the response resp has $(6\lambda + \frac{31}{3}\log_2(p)) \approx (6\lambda + 11\log_2(p))$ bits. Table 2 illustrates the signature sizes based on Section 3.3 under the B-SIDH setup [19].

$\log_2(p)$	λ	κ	Security Level	Private key	Public Key	Signature
377	128	219	NIST Level 1	24 B	96 B	84.19 kB
546	192	329	NIST Level 3	35 B	138 B	183.72 kB
697	256	438	NIST Level 5	44 B	176 B	315.53 kB
434	128	219	NIST Level 1	28 B	110 B	93.40 kB
503	160	274	NIST Level 2	32 B	126 B	136.58 kB
610	192	329	NIST Level 3	39 B	154 B	199.53 kB
751	256	438	NIST Level 5	47 B	188 B	331.32 kB

Table 1: Signature sizes correspond with the average case. Private keys correspond to integer coefficients sk in \mathbb{Z}_A , while public keys are elliptic curves $E: y^2 = x^3 + Ax^2 + x$ described by the element A in \mathbb{F}_{p^2} . Since the isogeny degrees satisfy $A, B \approx \sqrt{p}$, public keys are 4x larger than private keys.

$\log_2(p)$	λ	κ	Security Level	Private key	Public Key	Signature
256	128	219	NIST Level 1	32 B	64 B	98.18 kB
384	192	329	NIST Level 3	48 B	96 B	221.18 kB
512	256	438	NIST Level 5	64 B	128 B	392.58 kB

Table 2: Signature sizes correspond with the average case. Private keys correspond to integer coefficients sk in \mathbb{Z}_A , while public keys are elliptic curves $E: y^2 = x^3 + Ax^2 + x$ described by the element A in \mathbb{F}_{p^2} . Since the isogeny degrees satisfy $A, B \approx p$, public keys are 2x larger than private keys.

3.5 Over the Jacobian of genus-two curves

Following G2SIDH construction [31,38], we have another way to suggest sizes by working with Jacobian of genus two hyperelliptic curves³. This time the idea is to replace A -isogenies and B -isogenies with (A, A) -isogenies and (B, B) -isogenies. One crucial difference between SIDH and G2SIDH is that we do not have only two generators for the torsion subgroups; we have four generators instead, and two elements generate the isogeny kernels. For instance, given a public (A, A) -isogenous Jacobian J_1 to J_0 . This time Peggy wants to convince Victor that she knows the secret (A, A) -isogeny $\phi: J_0 \rightarrow J_1$, which implies knowing $\ker \phi = \langle K_{\phi,0}, K_{\phi,1} \rangle$. Here, J_0 is a public and fixed Jacobian of a genus two curve H_0 , and J_1 (the public key) comes from a genus two hyperelliptic curve H_1 . This time, the fixed degree relation over genus-two curves is defined as in Equation (3).

$$\mathcal{R}_{\text{deg}} := \{(J_0, J_1, d, \omega) \mid \omega: J_0 \rightarrow J_1 \text{ is a } (d, d)\text{-isogeny}\}. \quad (3)$$

³ For a deeper understanding of isogenies in the context of G2SIDH, we strongly suggest reading [31,13,38,37]

Similarly to Section 3.4, [21, Theorem 4] also extends if we repeat κ times the Σ -protocol described in [21, §5.3] and replace A -isogenies and B -isogenies with (A, A) -isogenies and (B, B) -isogenies, respectively. It becomes an SHVZK PoK with soundness $(2/3)^\kappa$, assuming the G2DSPP (described by Definition 3) is computationally hard.

Definition 3 (Genus two Decisional Supersingular Product Problem (G2DSPP): Alice’s case). *Let J_0 be a Jacobian of genus two curve H_0 as in the G2SIDH setting. Given a (A, A) -isogeny $\phi: J_0 \rightarrow J_1$ with kernel (A, A) -subgroup $\langle K_{\phi,0}, K_{\phi,1} \rangle$, the Genus two Decisional Supersingular Product Problem, labeled as G2DSPP, asks to distinguish between the following two distributions:*

- (J_2, J_3, ϕ') is the bottom of a random G2SIDH-square. That is, for a randomly chosen kernel (B, B) -subgroup $\langle K_{\psi,0}, K_{\psi,1} \rangle$, we have J_2 is the codomain of the (B, B) -isogeny ψ with kernel $\langle K_{\psi,0}, K_{\psi,1} \rangle$, J_3 is the codomain of the (B, B) -isogeny ψ' with kernel $\langle \phi(K_{\psi,0}), \phi(K_{\psi,1}) \rangle$, and $\phi': J_2 \rightarrow J_3$ is the (A, A) -isogeny with kernel $\langle \psi(K_{\phi,0}), \psi(K_{\phi,1}) \rangle$.
- (J_2, J_3, ϕ') such that J_2 is a randomly chosen Jacobian with same cardinality as J_0 , and $\phi': J_2 \rightarrow J_3$ is a random (A, A) -isogeny with kernel $\langle R_0, R_1 \rangle$ for some (A, A) -subgroup $\langle R_0, R_1 \rangle \in J_2[A]$.

Essentially, the genus-two recoverable Σ -protocol remains the same flow as in Section 3.3, but we need to consider that it requires double generators and isogeny evaluations, and genus two hyperelliptic curves $H: y^2 = f(x)$ are described by the degree-6 polynomial $f(x)$ over \mathbb{F}_{p^2} (but one can work with curve equations described by only three quadratic field coefficients r, s , and t [9,11]). Additionally, we have that the kernel generators of the (A, A) -isogenies and (B, B) -isogenies can be expressed by linear combinations determined with three integer coefficients c, d , and e of $\frac{\log_2(p)}{2}$ -bits. In summary, we need four times more of $\frac{\log_2(p)}{2}$ -bits integer coefficients to represent com_2 and com_3 , and three coefficients to represent the kernel generators of ϕ' , $\hat{\psi}$ and $\hat{\psi}'$. To be more precise, if $\text{chall} = 1$, then resp has $\frac{12\lambda+43\log_2(p)}{2}$ -bits. Otherwise, we have resp of $\frac{12\lambda+31\log_2(p)}{2}$ -bits. Consequently, on average we get a response resp with $\frac{36\lambda+105\log_2(p)}{6} \approx (6\lambda + 18\log_2(p))$ -bits. Since the best algorithm to find an isogeny is $\tilde{O}(p)$ (classically) and $\tilde{O}(\sqrt{p})$ (quantumly) [20], we can work with primes of 128 (NIST Level 1), 192 (NIST Level 3), and 256 (NIST Level 5). Table 3 lists the expected sizes of the signature over genus two curves.

Acknowledgements. We thank Víctor Mateu and Lucas Pandolfo Perin for encouraging us to make this work public. We thank Luca De Feo, Steven D. Galbraith, Jana Sotakova, Kaizhan Lin, and anonymous reviewers for their comments and discussions on [16] that we consider when writing this paper. We thank Andrea Basso for his comments about optimizing sizes when challenge chall is one. Finally, we thank Mukul Kulkarni for his thorough discussion and comments on an early version of this work.

$\log_2(p)$	λ	κ	Security Level	Private key	Public Key	Signature
128	128	219	NIST Level 1	24 B	96 B	84.13 kB
192	192	329	NIST Level 3	36 B	144 B	189.55 kB
256	256	438	NIST Level 5	48 B	192 B	336.45 kB

Table 3: Theoretical signature sizes. Private keys correspond to 3-tuples of integer coefficients $(\text{sk}_c, \text{sk}_d, \text{sk}_e)$ in \mathbb{Z}_A^3 , while public keys are genus two hyperelliptic curves $H: y^2 = f(x)$ with a degree-6 polynomial $f(x)$ over \mathbb{F}_{p^2} determined by three \mathbb{F}_{p^2} -elements [9,11]. Since the isogeny degrees satisfy $A, B \approx \sqrt{p}$, public keys are 4x larger than private keys.

References

1. Azarderakhsh, R., Campagna, M., Costello, C., De Feo, L., Hess, B., Jalali, A., Jao, D., Koziel, B., LaMacchia, B., Longa, P., Naehrig, M., Pereira, G., Renes, J., Soukharev, V., Urbanik, D.: Supersingular Isogeny Key Encapsulation. Third Round Candidate of the NIST’s post-quantum cryptography standardization process (2020), available at: <https://sike.org/>
2. Azarderakhsh, R., Jao, D., Kalach, K., Koziel, B., Leonardi, C.: Key compression for isogeny-based cryptosystems. In: Proceedings of the 3rd ACM International Workshop on ASIA Public-Key Cryptography. p. 1–10. AsiaPKC ’16, Association for Computing Machinery, New York, NY, USA (2016). <https://doi.org/10.1145/2898420.2898421>, <https://doi.org/10.1145/2898420.2898421>
3. Basso, A., Kutas, P., Merz, S.P., Petit, C., Weitkämper, C.: On adaptive attacks against jao-urbanik’s isogeny-based protocol. In: Nitaj, A., Youssef, A.M. (eds.) AFRICACRYPT 20. LNCS, vol. 12174, pp. 195–213. Springer, Heidelberg (Jul 2020). https://doi.org/10.1007/978-3-030-51938-4_10
4. Bernhard, D., Pereira, O., Warinschi, B.: How not to prove yourself: Pitfalls of the Fiat-Shamir heuristic and applications to Helios. In: Wang, X., Sako, K. (eds.) ASIACRYPT 2012. LNCS, vol. 7658, pp. 626–643. Springer, Heidelberg (Dec 2012). https://doi.org/10.1007/978-3-642-34961-4_38
5. Beullens, W.: Week 4: Signatures based on SIDH and CSIDH. Isogeny-based cryptography school pp. 1–23 (2021), https://homes.esat.kuleuven.be/~wbeullen/week4_1.pdf, last online access on June 1st, 2022: https://homes.esat.kuleuven.be/~wbeullen/week4_1.pdf
6. Beullens, W., Kleinjung, T., Vercauteren, F.: CSI-FiSh: Efficient isogeny based signatures through class group computations. In: Galbraith, S.D., Moriai, S. (eds.) ASIACRYPT 2019, Part I. LNCS, vol. 11921, pp. 227–247. Springer, Heidelberg (Dec 2019). https://doi.org/10.1007/978-3-030-34578-5_9
7. Biasse, J., Bonnetain, X., Pring, B., Schrottenloher, A., Youmans, W.: A trade-off between classical and quantum circuit size for an attack against CSIDH. Journal of Mathematical Cryptology **15**(1), 4–17 (2021). <https://doi.org/10.1515/jmc-2020-0070>
8. Bonnetain, X., Schrottenloher, A.: Quantum security analysis of CSIDH. In: Canteaut and Ishai [10], pp. 493–522. https://doi.org/10.1007/978-3-030-45724-2_17
9. Bruin, N., Flynn, E.V., Testa, D.: Descent via (3,3)-isogeny on jacobians of genus 2 curves. arXiv preprint arXiv:1401.0580 (2014), <https://arxiv.org/abs/1401.0580>

10. Canteaut, A., Ishai, Y. (eds.): EUROCRYPT 2020, Part II, LNCS, vol. 12106. Springer, Heidelberg (May 2020)
11. Castryck, W., Decru, T.: Multiradical isogenies. Cryptology ePrint Archive, Report 2021/1133 (2021), <https://eprint.iacr.org/2021/1133>
12. Castryck, W., Decru, T.: An efficient key recovery attack on SIDH. In: Hazay and Stam [35], pp. 423–447. https://doi.org/10.1007/978-3-031-30589-4_15
13. Castryck, W., Decru, T., Smith, B.: Hash functions from superspecial genus-2 curves using richelot isogenies. *J. Math. Cryptol.* **14**(1), 268–292 (2020). <https://doi.org/10.1515/jmc-2019-0021>
14. Castryck, W., Lange, T., Martindale, C., Panny, L., Renes, J.: CSIDH: An efficient post-quantum commutative group action. In: Peyrin, T., Galbraith, S. (eds.) ASIACRYPT 2018, Part III. LNCS, vol. 11274, pp. 395–427. Springer, Heidelberg (Dec 2018). https://doi.org/10.1007/978-3-030-03332-3_15
15. Chávez-Saab, J., Chi-Domínguez, J.J., Jaques, S., Rodríguez-Henríquez, F.: The SQALE of CSIDH: sublinear Vélu quantum-resistant isogeny action with low exponents. *Journal of Cryptographic Engineering* **12**(3), 349–368 (Sep 2022). <https://doi.org/10.1007/s13389-021-00271-w>
16. Chi-Domínguez, J., Mateu, V., Perin, L.P.: SIDH-sign: an efficient SIDH PoK-based signature (2022), <https://eprint.iacr.org/2022/475>
17. Costello, C.: B-SIDH: Supersingular isogeny Diffie-Hellman using twisted torsion. In: Moriai, S., Wang, H. (eds.) ASIACRYPT 2020, Part II. LNCS, vol. 12492, pp. 440–463. Springer, Heidelberg (Dec 2020). https://doi.org/10.1007/978-3-030-64834-3_15
18. Costello, C., Jao, D., Longa, P., Naehrig, M., Renes, J., Urbanik, D.: Efficient compression of SIDH public keys. In: Coron, J.S., Nielsen, J.B. (eds.) EUROCRYPT 2017, Part I. LNCS, vol. 10210, pp. 679–706. Springer, Heidelberg (Apr / May 2017). https://doi.org/10.1007/978-3-319-56620-7_24
19. Costello, C., Meyer, M., Naehrig, M.: Sieving for twin smooth integers with solutions to the prouhet-tarry-escott problem. In: Canteaut, A., Standaert, F.X. (eds.) EUROCRYPT 2021, Part I. LNCS, vol. 12696, pp. 272–301. Springer, Heidelberg (Oct 2021). https://doi.org/10.1007/978-3-030-77870-5_10
20. Costello, C., Smith, B.: The supersingular isogeny problem in genus 2 and beyond. In: Ding, J., Tillich, J.P. (eds.) Post-Quantum Cryptography - 11th International Conference, PQCrypto 2020. pp. 151–168. Springer, Heidelberg (2020). https://doi.org/10.1007/978-3-030-44223-1_9
21. De Feo, L., Dobson, S., Galbraith, S.D., Zobernig, L.: SIDH proof of knowledge. In: Agrawal, S., Lin, D. (eds.) ASIACRYPT 2022, Part II. LNCS, vol. 13792, pp. 310–339. Springer, Heidelberg (Dec 2022). https://doi.org/10.1007/978-3-031-22966-4_11
22. De Feo, L., Galbraith, S.D.: SeaSign: Compact isogeny signatures from class group actions. In: Ishai, Y., Rijmen, V. (eds.) EUROCRYPT 2019, Part III. LNCS, vol. 11478, pp. 759–789. Springer, Heidelberg (May 2019). https://doi.org/10.1007/978-3-030-17659-4_26
23. De Feo, L., Jao, D., Plût, J.: Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. *Journal of Mathematical Cryptology* **8**(3), 209–247 (2014). <https://doi.org/10.1515/jmc-2012-0015>
24. De Feo, L., Kohel, D., Leroux, A., Petit, C., Wesolowski, B.: SQISign: Compact post-quantum signatures from quaternions and isogenies. In: Moriai, S., Wang, H. (eds.) ASIACRYPT 2020, Part I. LNCS, vol. 12491, pp. 64–93. Springer, Heidelberg (Dec 2020). https://doi.org/10.1007/978-3-030-64837-4_3

25. De Feo, L., Leroux, A., Longa, P., Wesolowski, B.: New algorithms for the deuring correspondence - towards practical and secure SQISign signatures. In: Hazay and Stam [35], pp. 659–690. https://doi.org/10.1007/978-3-031-30589-4_23
26. Decru, T., Panny, L., Vercauteren, F.: Faster SeaSign signatures through improved rejection sampling. In: Ding and Steinwandt [27], pp. 271–285. https://doi.org/10.1007/978-3-030-25510-7_15
27. Ding, J., Steinwandt, R. (eds.): Post-Quantum Cryptography - 10th International Conference, PQCrypto 2019. Springer, Heidelberg (2019)
28. Dobson, S., Galbraith, S.D., LeGrow, J.T., Ti, Y.B., Zobernig, L.: An adaptive attack on 2-SIDH. *International Journal of Computer Mathematics: Computer Systems Theory* **5**(4), 282–299 (2020). <https://doi.org/10.1080/23799927.2020.1822446>
29. El Kaafarani, A., Katsumata, S., Pintore, F.: Lossy CSI-FiSh: Efficient signature scheme with tight reduction to decisional CSIDH-512. In: Kiayias, A., Kohlweiss, M., Wallden, P., Zikas, V. (eds.) PKC 2020, Part II. LNCS, vol. 12111, pp. 157–186. Springer, Heidelberg (May 2020). https://doi.org/10.1007/978-3-030-45388-6_6
30. Fiat, A., Shamir, A.: How to prove yourself: Practical solutions to identification and signature problems. In: Odlyzko, A.M. (ed.) CRYPTO’86. LNCS, vol. 263, pp. 186–194. Springer, Heidelberg (Aug 1987). https://doi.org/10.1007/3-540-47721-7_12
31. Flynn, E.V., Ti, Y.B.: Genus two isogeny cryptography. In: Ding and Steinwandt [27], pp. 286–306. https://doi.org/10.1007/978-3-030-25510-7_16
32. Fouotsa, T.B., Petit, C.: A new adaptive attack on SIDH. In: Galbraith, S.D. (ed.) CT-RSA 2022. LNCS, vol. 13161, pp. 322–344. Springer, Heidelberg (Mar 2022). https://doi.org/10.1007/978-3-030-95312-6_14
33. Galbraith, S.D., Petit, C., Shani, B., Ti, Y.B.: On the security of supersingular isogeny cryptosystems. In: Cheon, J.H., Takagi, T. (eds.) ASIACRYPT 2016, Part I. LNCS, vol. 10031, pp. 63–91. Springer, Heidelberg (Dec 2016). https://doi.org/10.1007/978-3-662-53887-6_3
34. Galbraith, S.D., Petit, C., Silva, J.: Identification protocols and signature schemes based on supersingular isogeny problems. *Journal of Cryptology* **33**(1), 130–175 (Jan 2020). <https://doi.org/10.1007/s00145-019-09316-0>
35. Hazay, C., Stam, M. (eds.): EUROCRYPT 2023, Part V, LNCS, vol. 14008. Springer, Heidelberg (Apr 2023)
36. Jao, D., De Feo, L.: Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. In: Yang, B.Y. (ed.) Post-Quantum Cryptography - 4th International Workshop, PQCrypto 2011. pp. 19–34. Springer, Heidelberg (Nov / Dec 2011). https://doi.org/10.1007/978-3-642-25405-5_2
37. Kunzweiler, S.: Efficient computation of $(2^n, 2^n)$ -isogenies. *Cryptology ePrint Archive*, Report 2022/990 (2022), <https://eprint.iacr.org/2022/990>
38. Kunzweiler, S., Ti, Y.B., Weitkämper, C.: Secret keys in genus-2 SIDH. In: Al-Tawy, R., Hülsing, A. (eds.) SAC 2021. LNCS, vol. 13203, pp. 483–507. Springer, Heidelberg (Sep / Oct 2022). https://doi.org/10.1007/978-3-030-99277-4_23
39. Leonardi, C.: A note on the ending elliptic curve in SIDH. *Cryptology ePrint Archive*, Report 2020/262 (2020), <https://eprint.iacr.org/2020/262>
40. Longa, P.: Efficient algorithms for large prime characteristic fields and their application to bilinear pairings and supersingular isogeny-based protocols. *Cryptology ePrint Archive*, Report 2022/367 (2022), <https://eprint.iacr.org/2022/367>
41. Maino, L., Martindale, C.: An attack on SIDH with arbitrary starting curve. *Cryptology ePrint Archive*, Report 2022/1026 (2022), <https://eprint.iacr.org/2022/1026>

42. Oudompheng, R., Pope, G.: A note on reimplementing the castryck-decru attack and lessons learned for SageMath. Cryptology ePrint Archive, Report 2022/1283 (2022), <https://eprint.iacr.org/2022/1283>
43. Peikert, C.: He gives C-sieves on the CSIDH. In: Canteaut and Ishai [10], pp. 463–492. https://doi.org/10.1007/978-3-030-45724-2_16
44. Robert, D.: Breaking SIDH in polynomial time. Cryptology ePrint Archive, Report 2022/1038 (2022), <https://eprint.iacr.org/2022/1038>
45. Unruh, D.: Post-quantum security of Fiat-Shamir. In: Takagi, T., Peyrin, T. (eds.) ASIACRYPT 2017, Part I. LNCS, vol. 10624, pp. 65–95. Springer, Heidelberg (Dec 2017). https://doi.org/10.1007/978-3-319-70694-8_3
46. Yoo, Y., Azarderakhsh, R., Jalali, A., Jao, D., Soukharev, V.: A post-quantum digital signature scheme based on supersingular isogenies. In: Kiayias, A. (ed.) FC 2017. LNCS, vol. 10322, pp. 163–181. Springer, Heidelberg (Apr 2017)

Appendix

A Security Proofs

A.1 Proof of Lemma 2

We restate the Lemma 2 below and follow it by its proof.

Lemma 2 (Knowledge Soundness). *The optimized protocol in Figure 3 is knowledge sound with knowledge error $\frac{2}{3}$.*

Proof (Proof of Lemma 2). We will first prove the following useful lemma in order to prove the knowledge soundness of our protocol,

Lemma 4 (3-special soundness). *The protocol shown in Figure 3 is 3-special sound.*

Proof (Proof of Lemma 4). **3-special soundness.**

Let $((\text{com}_{\mathcal{H}}, -1, \text{resp}_{-1}))$, $((\text{com}_{\mathcal{H}}, 0, \text{resp}_0))$, and $((\text{com}_{\mathcal{H}}, 1, \text{resp}_1))$ be the three accepting transcripts. It is worth highlighting that the extractor cannot compute the bottom A-isogeny $\phi': E_2 \rightarrow E_3$ without using resp_1 .

The extractor deterministically calculates $(P, Q) \leftarrow \text{CanonicalBasis}_{\mathbb{E}}(E_2)$. From resp_1 , the extractor knows the values of $(E_2, \alpha_{P_2}, \beta_{P_2})$, $(E_3, \alpha_{Q_2}, \beta_{Q_2})$. Moreover, the extractor can therefore obtain

$$\begin{aligned} P_2 &= [\alpha_{P_2}]P + [\beta_{P_2}]Q, \\ Q_2 &= [\alpha_{Q_2}]P + [\beta_{Q_2}]Q, \\ P_3 &= \phi'(P_2), \text{ and} \\ Q_3 &= \phi'(Q_2). \end{aligned}$$

Next, the extractor can get (b, Δ) from any of the responses resp_{-1} or resp_0 . From these values of (P_2, Q_2) , (P_3, Q_3) and (b, Δ) the extractor can now compute the kernel generator $K_{\widehat{\psi}}$ and $K_{\widehat{\psi}'}$ with the help of $(P_2, Q_2, (b, \Delta))$ and $(P_3, Q_3, (b, \Delta))$ as in Section 3.1 vertical case, respectively. Additionally, the extractor can compute the isogenies ψ and $\widehat{\psi}'$. All the above calculations can be done in polynomial time in the variable $n = \log_2(p) = O(\lambda)$. Since the extractor has computed the isogenies ψ , ϕ' , and $\widehat{\psi}'$ we can now apply Lemma 1 and obtain the secret isogeny ϕ . Note that all the values obtained by the extractor from various responses are verified by checking the various commit values in com . Since, the transcripts are accepting, we know that these checks are passed successfully. Therefore, based on the computational binding property of the commitment scheme \mathbb{C} , it is guaranteed that extractor obtains the same values as committed in the first message. \square

It is important to note that if the extractor defined above receives the transcript, $((\text{com}_{\mathcal{H}}, 1, \text{resp}_1))$ then it can extract a valid witness as explained above even if it get only *one* more transcript as $((\text{com}_{\mathcal{H}}, c, \text{resp}_c))$ where $c \in$

$\{0, -1\}$. However, this is not sufficient to achieve the 2-special soundness property (which would be desirable) since the transcripts $((\text{com}_{\mathcal{H}}, 0, \text{resp}_0))$ and $((\text{com}_{\mathcal{H}}, -1, \text{resp}_{-1}))$ do not yield a valid witness.

Since the optimized protocol is 3-special sound and the total number of possible challenges in the challenge space is equal to 3. This gives a knowledge soundness error of $\frac{2}{3}$. \square

A.2 Proof of Lemma 3

We restate the Lemma 3 below and follow it by its proof.

Lemma 3 (Special Honest Verifier Zero-Knowledge). *Let \mathcal{C} be a statistically hiding and computationally binding commitment scheme. Assuming the hardness of DSPP the optimized protocol in Figure 3 is computationally SHVZK in the random oracle model.*

Proof (Proof of Lemma 3). We define an efficient simulator which produces transcripts which are indistinguishable from the real transcripts produced by an honest execution of the protocol in Figure 3. On input $(E_0, \text{pk} = E_1)$ the simulator begins by sampling the challenge $\text{chall} \leftarrow \{-1, 0, 1\}$ uniformly at random. The simulator proceeds based on the sampled challenge chall as follows:

- **If $\text{chall} = -1$:** Simulator begins with sampling a random order- \mathbb{B} kernel generator K_ψ in E_0 and then it chooses a random basis $\{P_2, Q_2\}$ of $E_2[\mathbb{B}]$. It then computes (com_2, r_L) and $((c, d), r)$ honestly (as in the `commitment` procedure from Section 2.2). Simulator also computes $(\alpha_{P_2}, \beta_{P_2})$ and (b, Δ) honestly (as described in Section 3.1). It then computes the commitments $\text{com}_L = \mathcal{C}(\text{com}_2 \parallel r_L)$ and $\text{com}' = \mathcal{C}((b, \Delta) \parallel r)$ and sets com_R to uniform random value (chosen from the appropriate domain). Simulator sets the commitment $\text{com} = (\text{com}_L, \text{com}_R, \text{com}')$ and sets the response as $\text{resp}_{-1} = (\text{com}_R, E_2, (\alpha_{P_2}, \beta_{P_2}), (\alpha_{Q_2}, \beta_{Q_2}), r_L, (b, \Delta), r)$. It then outputs the transcript as $(\text{com}, -1, \text{resp}_{-1})$. The transcript is valid (accepting) since all the values which are verified when $\text{chall} = -1$ are computed honestly. The only value that is not computed honestly, and hence is not identically distributed to the corresponding values in the real honest execution of the protocol, is com_R . However, this is indistinguishable since we assume that the commitment scheme \mathcal{C} is statistically hiding.
- **If $\text{chall} = 0$:** In this case, the Simulator's strategy is analogous to the case when $\text{chall} = -1$. Simulator samples a random order- \mathbb{B} kernel generator $K_{\psi'}$ in E_1 . It also chooses a random basis $\{P_3, Q_3\}$ of $E_3[\mathbb{B}]$. Simulator computes (com_3, r_R) and $((c, d), r)$ honestly (as in the `commitment` procedure from Section 2.2). It then computes $(\alpha_{P_3}, \beta_{P_3})$ and (b, Δ) honestly (as described in Section 3.1). Next, Simulator computes $\text{com}_R = \mathcal{C}(\text{com}_3 \parallel r_R)$ and $\text{com}' = \mathcal{C}((b, \Delta) \parallel r)$ and sets com_L to uniform random value (chosen from the appropriate domain). Simulator sets $\text{com} = (\text{com}_L, \text{com}_R, \text{com}')$ and sets $\text{resp}_0 = (\text{com}_L, E_3, (\alpha_{P_3}, \beta_{P_3}), (\alpha_{Q_3}, \beta_{Q_3}), r_R, (b, \Delta), r)$. It then outputs the transcript as $(\text{com}, 0, \text{resp}_0)$. The transcript is valid (accepting) since

all the values which are verified when $\mathbf{chall} = 0$ are computed honestly. The only value that is not computed honestly, and hence is not identically distributed to the corresponding values in the real honest execution of the protocol, is \mathbf{com}_L . However, this is indistinguishable since we assume that the commitment scheme \mathbf{C} is statistically hiding.

- **If $\mathbf{chall} = 1$:** In this case, the simulator selects a random supersingular elliptic curve E_2 . It then samples a uniform random kernel generator $K_{\phi'}$ of order- A in E_2 . The simulator also computes the isogeny $\phi': E_2 \rightarrow E_3$. That is, the simulator computes a kernel generator $K_{\phi'}$ with the help of $\mathbf{CanonicalBasis}_A(E_2)$ (which outputs P', Q') and (b', Δ') as in Section 3.1 horizontal case.

It then generates a random basis $\{P_2, Q_2\}$ of $E_2[\mathbf{B}]$ and computes $P_3 = \phi'(P_2)$ and $Q_3 = \phi'(Q_2)$. Simulator then set $\mathbf{com}_2 = (E_2, P_2, Q_2)$ and creates the commitment, $\mathbf{com}_L = \mathbf{C}(\mathbf{com}_2 || r_L)$ for some r_L samples uniform randomly. It also computes, $\mathbf{com}_R = \mathbf{C}(\mathbf{com}_3 || r_R)$ where r_R is sampled uniformly at random and $\mathbf{com}_3 = (E_3, P_3, Q_3)$. The simulator sets \mathbf{com}' to uniform random value (chosen from the appropriate domain). Next, it sets the commitment as $\mathbf{com} = (\mathbf{com}_L, \mathbf{com}_R, \mathbf{com}')$ and the response as $\mathbf{resp}_1 = (\mathbf{com}', E_2, (\alpha_{P_2}, \beta_{P_2}), (\alpha_{Q_2}, \beta_{Q_2}), r_L, (b', \Delta'), E_3, r_R)$. It then outputs the transcript as $(\mathbf{com}, 1, \mathbf{resp}_1)$. As in the previous cases, the transcript is valid (accepting) since all the values which are verified when $\mathbf{chall} = 1$ are computed honestly. However, note that the curves E_2, E_3 , and the isogeny ϕ' are not distributed as in the real (honest) execution of the protocol. In fact, in the real protocol they are distributed as the first distribution in Definition 2, whereas the simulated values above are distributed as per the second distribution in Definition 2. Therefore, distinguishing between the real and simulated transcripts based on this information contradicts the assumption that DSPP is computationally hard problem. Additionally, another value that is not computed honestly, and hence is not identically distributed to the corresponding values in the real honest execution of the protocol, is \mathbf{com}' . However, this is indistinguishable since we assume that the commitment scheme \mathbf{C} is statistically hiding.

The above simulator runs in polynomial-time since it follows all the efficient computations of prover and samples uniform random values from well-defined finite domains. It also generates the transcripts which are indistinguishable from the real ones without knowing the secret isogeny $\phi: E_0 \rightarrow E_1$, this completes the proof. \square

B Size comparisons against isogeny-based signatures

As mentioned in Section 1, the short keys are the most attractive feature of isogeny-based signature construction. In contrast, such constructions have a high latency in practice, which seems to be much easier to improve. This section compares state-of-the-art isogeny-based signatures that remain secure against

Castrycck-Decru family attacks in terms of byte lengths. Currently, there are different families of isogeny-based Σ -protocols, such as:

- CSIDH-based: Sea-sign [22,26], CSI-FiSh [6] and the Lossy CSI-FiSh [29];
- SIDH-based: [21, §5.3]; and
- Quaternion-based: SQI-sign [24,25] and [34].

Since all CSIDH-based proposals are initially based over a 512-bits prime field, we compare them by moving into a 2048-bits prime field (as suggested in [8,43,15]). Using a 2048-bits CSIDH-prime impacts public-key sizes and timing efficiency; signature sizes stay fixed as in CSIDH-512. Now, due to the extended variety of CSIDH-based configurations determined by

- the number n of different isogeny degrees,
- the number B of isogenies per isogeny degree, and
- the number S of multiple public-key curves as CSIDH-base public-keys.

We try to englobe a fair comparison assuming $n = 74$, $B = 5$, and $S = 2^6$, which gives a good trade-off between small signature sizes and timings. We used the script from [26] to compute sizes of improved Sea-sign over a 2048-bits prime field. Table 4 lists all analyzed isogeny-based signature sizes in bytes.

Scheme	Private key	Public Key	Signature
[34, §4] with Fiat-Shamir transform	32 B	96 B	11.26 kB
Original SQI-sign [24,25]	16 B	64 B	204 B
Sea-sign [22]	16 B	16.384 kB	720 B
Sea-sign improvement from [26]	16 B	16.128 kB	7.22 kB
Simple variant of CSI-FiSh [6]	16 B	16.384 kB	560 B
Lossy CSI-FiSh [29]	16 B	16.896 kB	560 B
Optimized [21, §5.3] from Section 3.3 (p434)	28 B	110 B	93.40 kB
Optimized [21, §5.3] from Section 3.3 (p377)	24 B	96 B	84.19 kB
B-SIDH variant of [21, §5.3] from Section 3.4	32 B	64 B	98.18 kB
G2SIDH variant of [21, §5.3] from Section 3.5	24 B	96 B	84.13 kB

Table 4: Byte sizes concerning state-of-the-art isogeny-based signatures with close to NIST security Level 1. For a fair comparison, we set all CSIDH-based construction in [22,26,6,29] over a 2048-bits prime field (as suggested in [8,43,15]). Large CSIDH primes only impact public-key sizes and timing efficiency; signature sizes stay fixed as in CSIDH-512. All the sizes concerning this work are theoretically estimated.