SHealS and HealS: isogeny-based PKEs from a key validation method for SIDH

Tako Boris Fouotsa[1][0000−0003−1821−8406] and Christophe Petit[2,3][0000−0003−3482−6743]

[1] Università Degli Studi Roma Tre, Italy
takoboris.fouotsa@uniroma3.it
[2] Université Libre de Bruxelles, Belgium
[3] University of Birmingham's School of Computer Science, UK
christophe.f.petit@gmail.com

Abstract. In 2016, Galbraith et al. presented an adaptive attack on the SIDH key exchange protocol. In SIKE, one applies a variant of the Fujisaki-Okamoto transform to force Bob to reveal his encryption key to Alice, which Alice then uses to re-encrypt Bob's ciphertext and verify its validity. Therefore, Bob can not reuse his encryption keys. There have been two other proposed countermeasures enabling static-static private keys: k-SIDH and its variant by Jao and Urbanik. These countermeasures are relatively expensive since they consist in running multiple parallel instances of SIDH.
In this paper, firstly, we propose a new countermeasure to the GPST adaptive attack on SIDH. Our countermeasure does not require key disclosure as in SIKE, nor multiple parallel instances as in k-SIDH. We translate our countermeasure into a key validation method for SIDH-type schmes. Secondly, we use our key validation to design HealSIDH, an efficient SIDH-type static-static key interactive exchange protocol. Thirdly, we derive a PKE scheme SHealS using HealSIDH. SHealS uses larger primes compared to SIKE, has larger keys and ciphertexts, but only 4 isogenies are computed in a full execution of the scheme, as opposed to 5 isogenies in SIKE. We prove that SHealS is IND-CPA secure relying on a new assumption we introduce and we conjecture its IND-CCA security. We suggest HealS, a variant of SHealS using a smaller prime, providing smaller keys and ciphertexts.
As a result, HealSIDH is a practically efficient SIDH based (interactive) key exchange incorporating a "direct" countermeasure to the GPST adaptive attack.

Keywords: Post-quantum cryptography · SIDH · SIKE · adaptive attacks · HealSIDH · SHealS · HealS.

1 Introduction

The general isogeny computational problem is the following: given two isogenous elliptic curves E and E', compute an isogeny from E to E'. This hard

problem was used by J. M. Couveignes [8], Rostovtsev and Stolbunov [27] to design a key exchange protocol using ordinary isogenies, and by Charles, Goren and Lauter [5] to design a cryptographic hash function using supersingular isogenies. The CRS (Couveignes-Rostovtsev-Stolbunov) key exchange scheme is less practical in general and is vulnerable to a sub-exponential quantum attack [6].

In 2011, Jao and De Feo proposed SIDH [20] that uses isogenies of supersingular elliptic curves. SIDH is efficient and it is not vulnerable to the sub-exponential quantum attack presented in [6]. Nevertheless, a recent paper by Kutas et al. [21] proves that hidden-shift like attacks apply to variants of SIDH with considerably overstretched parameters. The isogeny computational problem underlying the security of SIDH is believed to be hard to break, even when using a quantum computer. SIKE [19] (which is the state of art implementation of SIDH [20,13]) is the only isogeny-based Key Encapsulation Mechanism (KEM) submitted to the NIST post-quantum standardization process. Even though SIKE is not the most efficient candidate among KEMs in this competition, SIKE provides the most compact keys and ciphertexts. This has certainly contributed to its selection for the third round of the competition as an alternate candidate [24].

Contrarily to the ordinary case where isogenies commute, supersingular isogenies do not commute in general. In order to solve this issue in SIDH, the images of some well-chosen torsion points through the secret isogeny are computed and included in the public keys.

In 2016, Galbraith et al. [17] exploited this supplementary information to develop adaptive attacks on SIDH when one party has a static secret key. The main idea of the attack is that Bob replaces the images of the torsion points in his public key by malicious ones and obtains some information on Alice's static secret when looking at the obtained shared secret. Repeating this process a polynomial number of times, Bob totally recovers Alice's private key.

In SIKE, the attack is avoided by applying a variant [18] of the Fujisaki-Okamoto transform [15]. This transform forces Bob to reveal his encryption key to Alice. Two countermeasures enabling static-static key exchange have been proposed: k-SIDH [1] and a variant by Jao and Urbanik [30]. These schemes essentially consist in running k^2 parallel instances of SIDH with each party having k SIDH private keys, hence each party computes about k^2 isogenies. In [11] and in [2], it is shown that variants of the adaptive attacks still apply to these schemes, and that the attacks are exponential in k in general. Hence one needs a relatively large k, say $k = 46$ as suggested by [11], for these schemes to be secure. For $k = 46$, about $46^2 = 2116$ isogenies are computed in k-SIDH, hence the scheme is arguably not practical. To the best of our knowledge, there exists no practically efficient method to counter the adaptive attack on SIDH without revealing the encryption key and using re-encryption to verify the validity of the ciphertext.

CSIDH [4] is the perfect post-quantum alternative to the classic Diffie-Hellman key exchange due to its analogy to the later primitive. Meanwhile, its quantum security has been considerably degraded recently [25], [3], [7] and remains to be precisely estimated. CSIDH was originally instantiated with a 512 bit prime,

but due to analysis of its actual quantum security, in [7] it is suggested to use primes of up to 4000 bits to achieve the NIST level 1 security. The increase of the prime size impacts the efficiency of the scheme.

Contributions. The contributions of this paper are fourfold.

Firstly, we propose a new countermeasure to the GPST adaptive attack on SIDH. The main idea is that Bob enable Alice to verify that his torsion points were honestly generated. Consider an SIDH setting, let $\phi_A : E_0 \to E_A$ and $\phi'_A : E_B \to E_{BA}$ be Alice's secret isogenies, $\phi_B : E_0 \to E_B$ and $\phi'_B : E_A \to E_{AB}$ be Bob's secret isogenies in an SIDH instance. In Section 3, we prove that if Bob publishes the action of ϕ_B on $E_0[\ell_A^{2e_A}]$ and that of ϕ'_B on $E_A[\ell_A^{2e_A}]$, then Alice can exploit this information to verify Bob's public key validity. Working with SIDH parameters where $p = \ell_A^{e_A}\ell_B^{e_B}f$, the torsion points of order $\ell_A^{2e_A}$ and $\ell_B^{2e_B}$ would be defined over extensions of $\mathbb{F}_{p^2}$ of degree roughly $\ell_A^{e_A}$ and $\ell_B^{e_B}$ respectively. We hence increase the field characteristic to $p = \ell_A^{2e_A}\ell_B^{2e_B}f - 1$ (where f is a small co-factor) such that the later torsion groups are defined over $\mathbb{F}_{p^2}$. Also, we set the starting curve E_0 to be a random supersingular curve with unknown endomorphism ring to avoid improved torsion points attacks. We hence obtain an efficient key validation method which does not require key disclosure and re-encryption, as it is the case in SIKE.

Secondly, we incorporate this key validation method into a key exchange scheme: HealSIDH (**Heal**ed **SIDH**). Let $p = \ell_A^{2e_A}\ell_B^{2e_B}f - 1$ as required by the countermeasure, let $\phi_A : E_0 \to E_A$, $\phi'_A : E_B \to E_{BA}$, and $\phi_B : E_0 \to E_B$, $\phi'_B : E_A \to E_{AB}$ be Alice's and Bob' secret isogenies respectively. Alice reveals the action of ϕ_A on $E_0[\ell_B^{2e_B}]$ and that of ϕ'_A on $E_B[\ell_B^{2e_B}]$. Analogously, Bob reveals the action of ϕ_B on $E_0[\ell_A^{2e_A}]$ and that of ϕ'_B on $E_A[\ell_A^{2e_A}]$. Revealing the action of ϕ'_A and ϕ'_B on torsion points implies revealing points on the shared curve $E_{AB} = E_{BA}$. To avoid this, each party canonically generates a basis of the corresponding subgroup and reveals the coordinates of the points in this canonical basis. HealSIDH is an order of magnitude more efficient compared to k-SIDH (the existing countermeasure to the adaptive attack on SIDH) since only four isogenies are computed in HealSIDH while more than k^2 (with $46 \leq k$) of them are computed in k-SIDH. The security of HealSIDH against key recovery relies on Problem 4 which is a variant of the Supersingular Isogeny Computational Diffie-Hellman Problem (SSICDHP), Problem 1.

Thirdly, we design a PKE scheme using HealSIDH. Our PKE scheme is named SHealS: **S**tatic-static key **Heal**ed **S**IKE. The idea in SHealS is to use the points to encrypt the plaintext, in such a way that the receiver solves a discrete logarithm problem in a group of smooth order to recover the plaintext. A similar idea is used in SiGamal [23] and SimS [14], but our design is different. SHealS uses primes two times larger (in terms of bit size) compared to SIKE primes, has larger keys and ciphertexts, but only 4 isogenies are computed and evaluated on torsion points in a full execution (KeyGeneration + Encryption + Decryption) of the scheme, as opposed to 5 isogenies in SIKE, among which 3 isogenies are evaluated on torsion points while the remaining two are not. For this reason, we believe SHealS efficiency is comparable to that of SIKE, but only an optimised

implementation of SHealS would help evaluate the exact timings and do a more precise efficiency comparison. The main advantage of SHealS over SIKE is the reuse of encryption keys. In fact, since there is no key disclosure, the encryption key can remain static for a given user. Moreover, this user can use this same key as a private key in the SHealS PKE setting. We prove that SHealS is IND-CPA secure relying on one new assumption we introduce. Despite not being able to come up with a succinct proof of IND-CCA security, we conjecture that SHealS is IND-CCA secure and provide arguments to support our conjecture.

Lastly, we suggest HealS, a variant of SHealS using a smaller prime, providing the same security level, smaller keys and ciphertexts. The size of the prime used in HealS is only 1.5 times that of the prime used in SIKE. This yields a speed-up over SHealS, smaller keys and ciphertexts; hence reducing the efficiency and key sizes gap between SHealS and SIKE. The drawback of HealS compared to SHealS is that private keys can not be used as encryption keys.

As a result, beside CSIDH whose quantum security remains to be precisely estimated, HealSIDH is a new efficient interactive post-quantum key exchange scheme enabling static-static key setting. Moreover, we believe the fact that there is no key disclosure in SHealS and HealS makes of them promising PKE schemes.

Related work. While this work was under submission, an SIDH Proof of Knowledge mechanism [12] was published online by De Feo et al. This mechanism enable any party in an SIDH instance to prove that his public key was honestly generated. The proof attached to the public key is obtained by performing an SIDH-type signature on the public key to to proof the knowledge of the secret isogeny and the correctness of the torsion points. For this reason, the proof is relatively large ($O(\lambda^2)$), computing and verifying the proof are relatively time consuming compared to our schemes. Nevertheless, their proof enables the design of an SIDH based NIKE while our key exchange HealSIDH is interactive.

Outline. The remaining of this paper is organized as follows: in Section 2, we recall some generalities about PKE schemes, elliptic curves and isogenies. We briefly present SIDH, the improved torsion points attacks and the GPST adaptive attack. We end Section 2 by describing existing countermeasures to the GPST adaptive attacks. Section 3 is devoted to our countermeasure. In Section 4 we present HealSIDH key exchange and in Section 5 we construct the SHealS PKE scheme. In Section 6, we provide a concrete instantiation of HealSIDH and SHealS, and provide a high level comparison to k-SIDH and SIKE respectively. In Section 7, we present HealS and in Section 8 we conclude the paper.

2 Preliminaries

2.1 Public key encryption

We recall standard security definitions related to public key encryption.

Definition 1 (PKE). *A Public Key Encryption scheme $\mathcal{P}_\lambda$ is a triple of PPT algorithms (*KeyGeneration*,* Encryption*,* Decryption*) that satisfy the following.*

1. *Given a security parameter λ as input, the key generation algorithm* KeyGeneration *outputs a public key pk, a private key sk and a plaintext space $\mathcal{M}$.*
2. *Given a plaintext $\mu \in \mathcal{M}$ and a public key pk as inputs, the encryption algorithm* Encryption *outputs a ciphertext $\mathsf{c} = \mathsf{Encryption}_{pk}(\mu)$.*
3. *Given a ciphertext c and sk as inputs, the decryption algorithm* Decryption *outputs a plain text $= \mathsf{Decryption}_{sk}(\mathsf{c})$.*

Definition 2 (Correctness). *A PKE scheme $\mathcal{P}_\lambda$ is correct if for any pair of keys (pk, sk) and for every plaintext $\mu \in \mathcal{M}$,*

$$\mathsf{Decryption}_{sk}\left(\mathsf{Encryption}_{pk}(\mu)\right) = \mu.$$

Definition 3 (IND-CPA secure). *A PKE scheme $\mathcal{P}_\lambda$ is IND-CPA secure if for every PPT adversary $\mathcal{A}$,*

$$Pr\left[b = b^* \left| \begin{array}{l} (pk, sk) \leftarrow \mathsf{KeyGeneration}(\lambda), \mu_0, \mu_1 \leftarrow \mathcal{M}, \\ b \xleftarrow{\$} \{0,1\}, \mathsf{c} \leftarrow \mathsf{Encryption}_{pk}(\mu_b), b^* \leftarrow \mathcal{A}(pk, \mathsf{c}) \end{array} \right.\right] = \frac{1}{2} + \mathsf{negl}(\lambda).$$

Definition 4 (IND-CCA secure). *A PKE scheme $\mathcal{P}_\lambda$ is IND-CCA secure if for every PPT adversary $\mathcal{A}$,*

$$Pr\left[b = b^* \left| \begin{array}{l} (pk, sk) \leftarrow \mathsf{KeyGeneration}(\lambda), \mu_0, \mu_1 \leftarrow \mathcal{A}^{O(\cdot)}(pk, \mathcal{M}), \\ b \xleftarrow{\$} \{0,1\}, \mathsf{c} \leftarrow \mathsf{Encryption}_{pk}(\mu_b), b^* \leftarrow \mathcal{A}^{O(\cdot)}(pk, \mathsf{c}) \end{array} \right.\right] = \frac{1}{2} + \mathsf{negl}(\lambda),$$

where $O(\cdot)$ is a decryption oracle that when given a ciphertext $\mathsf{c}' \neq \mathsf{c}$, outputs $\mathsf{Decryption}_{sk}(\mathsf{c}')$ or $\perp$ if the ciphertext c' is invalid.

2.2 Elliptic curves and isogenies

An elliptic curve is a rational smooth curve of genus one with a distinguished point at infinity. Elliptic curves can be seen as commutative groups with respect to a group addition having the point at infinity as neutral element. When an elliptic curve E is defined over a finite field $\mathbb{F}_q$, the set of $\mathbb{F}_q$-rational points $E(\mathbb{F}_q)$ of E is a subgroup of E. For every integer N coprime with q, the N-torsion subgroup $E[N]$ of E is isomorphic to $\mathbb{Z}_N \oplus \mathbb{Z}_N$.

An isogeny from E to E' is a rational map from E to E' which is also a group morphism. The kernel of an isogeny is always finite and entirely defines the isogeny up to powers of the Frobenius. Given a finite subgroup G of E, there exists a Frobenius free isogeny of domain E having kernel G, called a separable isogeny. Its degree is equal to the size of its kernel. The co-domain of this isogeny is denoted by E/G. The isogeny and the co-domain E/G can be computed from the knowledge of the kernel using Vélu's formulas [28] whose efficiency depends on the smoothness of the isogeny degree.

An endomorphism of an elliptic curve E is an isogeny from E to E. The group structure of E is closely related to that of its endomorphism ring. When E is defined over a finite field, the endomorphism ring of E is either an order in a quadratic field, in which case we say E is ordinary, or a maximal order in a quaternion algebra in which case we say E is supersingular. The generic isogeny problem is harder to solve for supersingular curves (for which the best attacks are exponential) than ordinary curves (for which there exists a sub-exponential attack [6]). SIDH is based on supersingular isogenies.

2.3 SIDH

The SIDH scheme is defined as follows.

Setup. Let $p = \ell_A^{e_A}\ell_B^{e_B} - 1$ be a prime such that $\ell_A^{e_A} \approx \ell_B^{e_B} \approx \sqrt{p}$. Let E_0 be a supersingular curve defined over $\mathbb{F}_{p^2}$. Set $E_0[\ell_A^{e_A}] = \langle P_A, Q_A\rangle$ and $E_0[\ell_B^{e_B}] = \langle P_B, Q_B\rangle$. The public parameters are E_0, p, ℓ_A, ℓ_B, e_A, e_B, P_A, Q_A, P_B, Q_B.

KeyGeneration. The secret key sk_A of Alice is a uniformly random integer α sampled from $\mathbb{Z}_{\ell_A^{e_A}}$. Compute the cyclic isogeny $\phi_A : E_0 \to E_A = E_0/\langle P_A + [\alpha]Q_A\rangle$. The public key of Alice is the tuple $\mathsf{pk}_A = (E_A, \phi_A(P_B), \phi_A(Q_B))$. Analogously, Bob's secret key sk_B is a uniformly random integer β sampled from $\mathbb{Z}_{\ell_B^{e_B}}$ and his public key is $\mathsf{pk}_B = (E_B, \phi_B(P_A), \phi_B(Q_A))$ where $\phi_B : E_0 \to E_B = E_0/\langle P_B + [\beta]Q_B\rangle$.

KeyExchange. Upon receiving (E_B, R_a, S_a), Alice checks that $e(R_a, S_a) = e(P_A, Q_A)^{\ell_B^{e_B}}$, if not she aborts. She computes the isogeny $\phi'_A : E_B \to E_{BA} = E_B/\langle R_a + [\alpha]S_a\rangle$. Her shared key is $j(E_{BA})$. Similarly, upon receiving (E_A, R_b, S_b), Bob checks that $e(R_b, S_b) = e(P_B, Q_B)^{\ell_A^{e_A}}$, if not he aborts. He computes the isogeny $\phi'_B : E_A \to E_{AB} = E_A/\langle R_b + [\beta]S_b\rangle$. His shared key is $j(E_{AB})$.

The correctness of the key exchange follows from the fact that

$$E_A/\langle \phi_A(P_B) + [\beta]\phi_A(Q_B)\rangle \simeq E_0/\langle P_A + [\alpha]Q_A, P_B + [\beta]Q_B\rangle \simeq E_B/\langle \phi_B(P_A) + [\alpha]\phi_B(Q_A)\rangle\,.$$

The security of the SIDH key exchange protocol against shared key recovery relies on Problem 1. Furthermore, Problem 2 states that it is difficult to distinguish the shared secret from a random supersingular elliptic curve.

Problem 1 (Supersingular Isogeny Computational Diffie-Hellman). Given E_0, P_A, Q_A, P_B, Q_B, E_A, $\phi_A(P_B)$, $\phi_A(Q_B)$, E_B, $\phi_B(P_A)$, $\phi_B(Q_A)$ (defined as in SIDH), compute E_{AB}.

Problem 2 (Supersingular Isogeny Decisional Diffie-Hellman). Given E_0, P_A, Q_A, P_B, Q_B, E_A, $\phi_A(P_B)$, $\phi_A(Q_B)$, E_B, $\phi_B(P_A)$, $\phi_B(Q_A)$ (defined as in SIDH) and a random supersingular curve E, distinguish between $E = E_{AB}$ and $E \neq E_{AB}$.

An IND-CPA secure PKE from SIDH. One canonically derives a PKE schemes from SIDH as follows. Let $H : \mathbb{F}_{p^2} \to \{0,1\}^n$ be a cryptographic hash function.

KeyGeneration. Alice generates her key pair exactly as in SIDH.

Encryption. Let m be a plaintext. Bob generates a random $\beta \in \mathbb{Z}_{\ell_B^{e_B}}$ and executes the SIDH key exchange using Alice's public key to obtain $\mathsf{c}_0 = (E_B, \phi_B(P_A), \phi_B(Q_A))$ and $j_{AB} = j(E_{AB})$. The ciphertext is $(\mathsf{c}_0, \mathsf{c}_1 = H(j_{AB}) \oplus \mathsf{m})$.

Decryption. Given a ciphertext $(\mathsf{c}_0, \mathsf{c}_1)$, Alice completes the underlying SIDH key exchange to obtain $j_{BA} = j(E_{BA})$ and recovers the plaintext $\mathsf{m} = \mathsf{c}_1 \oplus H(j_{BA})$.

The above scheme is IND-CPA secure assuming Problem 2 is hard [13], but it is not IND-CCA since it is vulnerable to the GPST adaptive attack [17] that we present later in Section 2.5.

2.4 Passive torsion point attacks on SIDH

The direct key recovery attack (attacking one party's secret key) in SIDH translates into solving the following *Supersingular Isogeny Problem*.

Problem 3. Let A and B be two integers such that $\gcd(A, B) = 1$. Let E_0 be a supersingular elliptic curve defined over $\mathbb{F}_{p^2}$. Set $E_0[B] = \{P, Q\}$ and let $\phi : E_0 \to E_A$ be a random isogeny of degree A. Given E_0, E_A, P, Q, $\phi(P)$ and $\phi(Q)$, compute ϕ.

The difference between Problem 3 and the general isogeny problem is the fact that the action of ϕ on the group $E_0[B]$ is revealed. In 2017, Petit [26] exploited these torsion point images to design an algorithm that solves Problem 3 for a certain choice of unbalanced ($A \ll B$) parameters when the endomorphism ring of the starting curve E_0 is public. Petit's attack has recently been considerably improved by de Quehen et al. [10]. We refer to [10] for more details.

To counter the attack in unbalanced SIDH instances, one sets the starting curve E_0 to be a random supersingular curve with unknown endomorphism ring. We don't know how to generate random supersingular elliptic curves for which the endomorphism ring is unknown (also to the party generating the curve). This is considered as an open problem [9]. Hence one generally relies on a trusted party to generate a random curve which is then used as a public parameter of the scheme. This will be the case for the schemes presented in this paper.

2.5 GPST adaptive attack

In SIDH [13] one does a pairing-based check on the torsion points $\phi_B(P_A)$ and $\phi_B(Q_A)$ returned by a potentially malicious Bob. Let E be a supersingular elliptic curve, let N be an integer and let μ_N be the group of N-roots of unity. Let $e_N : E[N] \times E[N] \to \mu_N$ be the Weil pairing [16]. Let $\phi : E \to E'$ be an isogeny of degree M, then for $P, Q \in E[N]$,

$$e_N(\phi(P), \phi(Q)) = e_N(P, Q)^M$$

where the first pairing is computed on E' and the second one on E.
In SIDH, given (E_B, R, S) returned by Bob as public key, Alice checks if

$$e_{\ell_A^{e_A}}(R,S) = e_{\ell_A^{e_A}}(P_A, Q_A)^{\ell_B^{e_B}}.$$

As we will see below, this verification does not assure that the points R, S were honestly generated. More precisely, the pairing verification does not capture the GPST adaptive attack.

The GPST adaptive attack. The main idea of the Galbraith et al. adaptive attack [17] is that if Bob manipulates the torsion points $\phi_B(P_A)$ and $\phi_B(Q_A)$ conveniently, then he can get some information about Alice's private key α given that he knows if the secret curve computed by Alice is equal to E_{AB} or not. Hence in the attack scenario, Bob needs to have access to the later information. This access is provided to Bob through a key exchange oracle:

$$O(E, R, S, E') \text{ which returns 1 if } j(E') = j(E/\langle R + [\alpha]S\rangle) \text{ and 0 otherwise}$$

If one supposes that $\ell_A = 2$ and $e_A = n$, then after each query, Bob recovers one bit of

$$\alpha = \alpha_0 + 2^1\alpha_1 + 2^2\alpha_2 + \cdots + 2^{n-1}\alpha_{n-1}.$$

The attack recovers the first $n-2$ bits of α using $n-2$ oracle queries, and it recovers the two remaining bits by brute force. We refer to [17] for more details.

2.6 Existing countermeasures to the GPST adaptive attacks

The previous section has highlighted the need for a "better" key validation method for SIDH-type schemes. We now present SIKE and k-SIDH, that are currently the two main countermeasures to the GPST attack on SIDH.

SIKE (Supersingular Isogeny Key Encapsulation). Our description is more general compared to that submitted to the third round of the NIST competition [19], and it does not include key compression features. In the following scheme, G, H and F are hash functions and n is an integer, we refer to [19] for more details.

Setup. As in SIDH.

KeyGeneration. Generate a secret key $\mathsf{sk} = \alpha \in \mathbb{Z}_{\ell_A^{e_A}}$ and a public key $\mathsf{pk} = (E_A, \phi_A(P_B), \phi_A(Q_B))$ as in SIDH. Sample a uniformly random integer $s \in \{0,1\}^n$ and return $(s, \mathsf{sk}, \mathsf{pk})$.

Encapsulation. Sample a uniformly random integer m from $\{0,1\}^n$. Compute $\beta = G(m||\mathsf{pk}) \in \mathbb{Z}_{\ell_B^{e_B}}$ and compute $\mathsf{c}_0 = (E_B, \phi_B(P_A), \phi_B(Q_A))$ and E_{AB} as in the SIDH, together with $\mathsf{c}_1 = F(j(E_{AB})) \oplus m$ and $K = H(m||(\mathsf{c}_0, \mathsf{c}_1))$ and return $((\mathsf{c}_0, \mathsf{c}_1), K)$.

Decapsulation. From $(\mathsf{c}_0, \mathsf{c}_1)$, compute E_{BA} as in SIDH and $m' = \mathsf{c}_1 \oplus F(j(E_{BA}))$.

Re-encrypt m' to obtain $\mathsf{c}'_0 = (E'_B, \psi_B(P_A), \psi_B(Q_A))$. If $\mathsf{c}_0 = \mathsf{c}'_0$, return $K = H(m'||(\mathsf{c}_0, \mathsf{c}_1))$, else return $K = H(s||(\mathsf{c}_0, \mathsf{c}_1))$.

In SIKE, the adaptive attacks are not applicable since during the decapsulation, Alice recomputes Bob's encryption key $\beta' = G(m'||\mathsf{pk}) \in \mathbb{Z}_{\ell_B^{e_B}}$ and checks if the obtained key leads to the curve and torsion points sent by Bob, this enables her to detect maliciously generated public keys. Therefore, the scheme requires key disclosure to the recipient. This is a common drawback to all post-quantum PKEs engaged in the NIST standardization process. In fact, as noticed in [1, §1], these schemes use ephemeral keys or indirect validation techniques that would expose one's key in the static-static setting.

Other countermeasures to the GPST attack. As a countermeasure to the GPST attack, Azarderakhsh et al. introduced k-SIDH [1]. In k-SIDH, Alice's private key is a tuple $\alpha = (\alpha_1, \cdots, \alpha_k) \in (\mathbb{Z}_{\ell_A^{e_A}})^k$ and Bob's private key is a tuple $\beta = (\beta_1, \cdots, \beta_k) \in (\mathbb{Z}_{\ell_B^{e_B}})^k$. Alice and Bob simultaneously run k^2 SIDH key exchange instances corresponding to the k^2 couples of Alice and Bob's SIDH private keys (α_i, β_j), $1 \leq i, j \leq k$. The shared secret is then obtained by applying a key derivation function to the corresponding k^2 SIDH shared secrets. The scheme quickly becomes impractical as k grows.

In [30], Jao and Urbanik propose a variant of k-SIDH that they expected to be more efficient. Their variant exploits non trivial automorphisms of the starting curve E_0 when this supersingular curve has j-invariant 0 or 1728 to reduce the number k of SIDH instances in k-SIDH. For example, in the case where the starting supersingular curve E_0 has j-invariant 0, there exists a non trivial automorphism η_6 of E_0 of order 6. Given a finite subgroup $G \subset E_0$, the curves E_0/G, $E_0/\eta_6(G)$ and $E_0/\eta_6^2(G)$, are isomorphic but it is not the case for the isogenies $E_0 \to E_0/G$, $E_0 \to E_0/\eta_6(G)$ and $E_0 \to E_0/\eta_6^2(G)$. Hence when performing a key exchange, these three isogenies will lead to three distinct SIDH shared keys. Hence with $\alpha' = (\alpha_1, \cdots, \alpha_{k'}) \in (\mathbb{Z}_{\ell_A^{e_A}})^{k'}$ and $\beta' = (\beta_1, \cdots, \beta_{k'}) \in (\mathbb{Z}_{\ell_B^{e_B}})^{k'}$, Alice and Bob can derive $3k'^2$ SIDH shared secrets contrarily to k'^2 for k-SIDH.

In [11], Dobson et al. show that the GPST attack can be adapted to k-SIDH. Nevertheless, the cost of the attack (number of queries to the key exchange oracle) grows exponentially with k. Dobson et al.'s attack does not directly apply to the Jao-Urbanik variant of k-SIDH. In [2], Basso et al. present an adaptation of this attack to the Jao-Urbanik variant. Moreover, they prove that considering their attack, for the same security level, k-SIDH is more efficient compared to the Jao-Urbanik variant. From these two attacks, one concludes that for k-SIDH and the Jao-Urbanik variant to be secure against adaptive attacks, one needs k to be relatively large ([11] suggests $k = 46$ for about 128 bits of security), consequently the schemes become less practical.

To sum up, as countermeasures to the GPST adaptive attack, SIKE imposes key disclosure while k-SIDH comes with a considerable efficiency drawback. We

address this in the next section by providing a new countermeasure which is more efficient compared to k-SIDH and without key disclosure.

3 A new countermeasure to the GPST adaptive attack

In this section, we describe a mechanism which enables Alice, when using a static key, to decide on the correctness of the torsion points returned by BoB. We translate this point correctness mechanism into a new key validation method.

3.1 Overview

In our scenario, like in SIKE, we suppose that the initiator of the communication (Bob) has to prove the validity of his torsion points to the other party (Alice). Let E_0, P_A, Q_A, P_B, Q_B, E_A, $\phi_A(P_B)$, $\phi_A(P_B)$ be the public parameters and Alice's public key in an SIDH instance. For simplicity, we suppose that the degree of Alice's isogeny is 2^a and that the degree of Bob's isogeny is 3^b for some integers a and b. In SIDH, Bob computes a cyclic isogeny $\phi_B : E_0 \to E_B$ of degree 3^b together with the images $\phi_B(P_A)$ and $\phi_B(Q_A)$ of P_A and Q_A. We say that the torsion points $R, S \in E_B[2^a]$ returned by Bob are *correct* if $R = [\lambda]\phi_B(P_A)$ and $S = [\lambda]\phi_B(Q_A)$ for some $\lambda \in \mathbb{Z}/2^a\mathbb{Z}^\times$. We establish a Points Correctness Verification (PCV) mechanism for Alice to determine if the torsion points computed by Bob are correct.

We start with an observation of Leonardi [22]: *"in an honest SIDH,* $\phi'_A \circ \phi_B = \phi'_B \circ \phi_A$*"*. Composing by $\widehat{\phi'_A}$ on the left, we get

$$[2^a] \circ \phi_B = \widehat{\phi'_A} \circ \phi'_B \circ \phi_A. \tag{1}$$

Let $P_2, Q_2 \in E_0[2^{2a}]$ be points such that $[2^a]P_2 = P_A$ and $[2^a]Q_2 = Q_A$. Then

$$\begin{cases} \phi'_A \circ \phi_B(P_2) = \phi'_B \circ \phi_A(P_2) \\ \phi'_A \circ \phi_B(Q_2) = \phi'_B \circ \phi_A(Q_2), \end{cases} \tag{2}$$

hence

$$\begin{cases} \phi_B(P_A) = \phi_B([2^2]P_2) = \widehat{\phi'_A} \circ \phi'_B \circ \phi_A(P_2) \\ \phi_B(Q_A) = \phi_B([2^a]Q_2) = \widehat{\phi'_A} \circ \phi'_B \circ \phi_A(Q_2) \end{cases} \tag{3}$$

Equation 3 suggests that if Alice can successfully check the equalities in Equation 2, then she can verify if Bob's torsion points are correct.

The idea of the PCV mechanism is that instead of revealing the action of $\phi_B : E_0 \to E_B$ on the 2^a-torsion sub-group of E_0, Bob reveals the action of ϕ_B on the 2^{2a}-torsion sub-group of E_0 and the action of $\phi'_B : E_A \to E_{AB}$ on the 2^{2a}-torsion sub-group of E_A. In our PCV mechanism, Bob's public key (when honestly computed) is $(E_B, \phi_B(P_2), \phi_B(Q_2))$. The action of $\phi'_B : E_A \to E_{AB}$ on the 2^{2a}-torsion sub-group of E_A is provided by canonically generating a new 2^{2a}-torsion basis $\{R_A, S_A\}$ of E_A and revealing $R_{ab} = \phi'_B(R_A)$ and $S_{ab} = \phi'_B(S_A)$.

At this point, Bob can be malicious in the following three ways:

1. honestly compute $R_a = \phi_B(P_2)$ and $S_a = \phi_B(Q_2)$, and maliciously compute $R_{ab} = \phi'_B(R_A)$ and $S_{ab} = \phi'_B(S_A)$;
2. maliciously compute $R_a = \phi_B(P_2)$ and $S_a = \phi_B(Q_2)$, and honestly compute $R_{ab} = \phi'_B(R_A)$ and $S_{ab} = \phi'_B(S_A)$;
3. maliciously compute $R_a = \phi_B(P_2)$ and $S_a = \phi_B(Q_2)$, and maliciously compute $R_{ab} = \phi'_B(R_A)$ and $S_{ab} = \phi'_B(S_A)$.

In the first two cases, we say that Bob is *partially point-malicious* and in the third case we say that Bob is *doubly point-malicious*.

Remark 1. We use the term *point-malicious* to highlight the fact that we focus only on the correctness of the torsion points outputted by BoB, not on the validity of the Bob's entire public key. Hence we are supposing that ϕ_B and ϕ'_B are cyclic isogenies of degree 3^b and only the torsion point were maliciously evaluated.

When Bob is partially point-malicious, then either the right hand term or the left hand term in Equation 2 is correctly computed by Alice. Hence the partial point-maliciousness of Bob would be detected since the other term of the equation would be different. Concretely, we have the following theorem.

Theorem 1. *Let E_0, P_A, Q_A, P_B, Q_B, E_A, $\phi_A(P_B)$, $\phi_A(P_B)$ be the public parameters and Alice's public key in an SIDH instance. Let P_2, $Q_2 \in E_0[2^{2a}]$ such that $[2^a]P_2 = P_A$ and $[2^a]Q_2 = Q_A$. Let (E_B, R_a, S_a) be Bob's public key. Moreover, let $\{R_A, S_A\}$ be a canonical basis of $E_A[2^{2a}]$ and let $\{R_{ab}, S_{ab}\}$ be its image through $\phi'_B : E_A \to E_{AB}$ outputted by Bob. Write $\phi_A(P_2) = [e_1]R_A + [f_1]S_A$ and $\phi_A(Q_2) = [e_2]R_A + [f_2]S_A$. Let us suppose that Bob is eventually partially point-malicious and let $\psi'_A : E_B \to E_B/\langle [2^a]R_a + [\alpha][2^a]S_a\rangle$ be the isogeny computed by Alice.*
If $e_{2^{2a}}(R_a, S_a) = e_{2^{2a}}(P_2, Q_2)^{3^b}$, $[e_1]R_{ab} + [f_1]S_{ab} = \psi'_A(R_a)$ and $[e_2]R_{ab} + [f_2]S_{ab} = \psi'_A(S_a)$, then Bob's torsion points are correct.

Proof. Noticing that $[e_1]R_{ab} + [f_1]S_{ab}$ stands for $\phi'_B \circ \phi_A(P_2)$ and $\psi'_A(R_a)$ for $\phi'_A \circ \phi_B(P_2)$, while $[e_2]R_{ab} + [f_2]S_{ab}$ stands for $\phi'_B \circ \phi_A(Q_2)$ and $\psi'_A(S_a)$ for $\phi'_A \circ \phi_B(Q_2)$, the theorem follows from the previous discussion. □

Remark 2. The points $\phi_A(P_2), \phi_A(Q_2) \in E_A[2^{2a}]$ are secret (known only by Alice). In fact their knowledge is equivalent to the knowledge of Alice's secret since $[2^a]P_2 = P_A$ and $[2^a]Q_2 = Q_A$.

For the third case where Bob is *doubly point-malicious*, we provide a more involved mathematical proof in the next paragraph.

3.2 The main theorem

In the previous section, we make use of points of order 2^{2a} or 3^{2b}. In SIDH parameters where $p = 2^a 3^b f - 1$, these points are defined over a large extension field of degree roughly $2^a \approx 3^b$. To make our key validation efficient, we use

primes of the form $p = 2^{2a}3^{2b}f - 1$. Moreover, we evaluate isogenies of degree 2^a on points of order $3^{2b} \approx 2^{2a}$. To avoid improved torsion points attacks or any variant of it, we set the starting curve E_0 to be a random supersingular curve with unknown endomorphism ring. Figure 1 summarizes the key validation mechanism hence obtained.

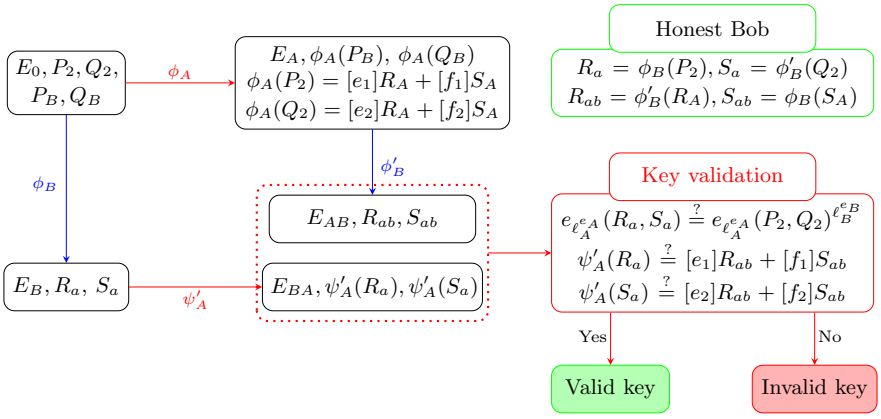

Fig. 1: Key validation mechanism for SIDH-type schemes. The curve E_0 is a random supersingular elliptic curve with unknown endomorphism ring defined over $\mathbb{F}_{p^2}$ where $p = 2^{2a}3^{2b}f - 1$.

We prove the following Theorem.

Theorem 2. *Let $p = 2^{2a}3^{2b}f - 1$ and let E_0 be a random supersingular elliptic curve with unknown endomorphism ring defined over $\mathbb{F}_{p^2}$. Let E_0, P_A, Q_A, P_B, Q_B, E_A, $\phi_A(P_B)$, $\phi_A(P_B)$ be the public parameters and Alice's public key in an SIDH instance. Let P_2, $Q_2 \in E_0[2^{2a}]$ such that $[2^a]P_2 = P_A$ and $[2^a]Q_2 = Q_A$. Let (E_B, R_a, S_a) be Bob's public key. Moreover, let $\{R_A, S_A\}$ be a canonical basis of $E_A[2^{2a}]$ and let $\{R_{ab}, S_{ab}\}$ be its image through $\phi'_B : E_A \to E_{AB}$ outputted by Bob. Write $\phi_A(P_2) = [e_1]R_A + [f_1]S_A$ and $\phi_A(Q_2) = [e_2]R_A + [f_2]S_A$. Let $\psi'_A : E_B \to E_B/\langle [2^a]R_a + [\alpha][2^a]S_a\rangle$ be the second isogeny computed by Alice during the key exchange.*

If $e_{2^{2a}}(R_a, S_a) = e_{2^{2a}}(P_2, Q_2)^{3^b}$, $[e_1]R_{ab} + [f_1]S_{ab} = \psi'_A(R_a)$ and $[e_2]R_{ab} + [f_2]S_{ab} = \psi'_A(S_a)$, then Bob's torsion points are correct.

Proof. Let us suppose that Bob is possibly doubly point-malicious, say

$$\begin{vmatrix} R_a = [x]\phi_B(P_2) + [y]\phi_B(Q_2) \\ S_a = [z]\phi_B(P_2) + [t]\phi_B(Q_2) \\ R_{ab} = [x']\phi'_B(R_A) + [y']\phi'_B(S_A) \\ S_{ab} = [z']\phi'_B(R_A) + [t']\phi'_B(S_A) \end{vmatrix}$$

for some integers x, y, z, t, x', y', z' and t' modulo 2^{2a}.

Let us suppose that $e_{2^{2a}}(R_a, S_a) = e_{2^{2a}}(P_2, Q_2)^{3^b}$, $[e_1]R_{ab}+[f_1]S_{ab} = \phi'_A(R_a)$ and $[e_2]R_{ab} + [f_2]S_{ab} = \phi'_A(S_a)$. We prove that $x = t = x' = t' = \pm 1$ and $y = z = y' = z' = 0$, which implies that Bob's torsion points are correct. Let

$$\phi'_A : E_B \to E_{BA} = E_B/\langle \phi_B(P_A) + [\alpha]\phi_B(Q_A)\rangle = E_B/\langle [2^a]\phi_B(P_2) + [\alpha][2^a]\phi_B(Q_2)\rangle$$

be the isogeny that ought to be computed by Alice if Bob's torsion points were correct and let

$$\psi'_A : E_B \to E_B/\langle [2^a]R_a + [\alpha][2^a]S_a\rangle = E_B/\langle \phi_B([a]P_A + [b]Q_A) + [\alpha]\phi_B([c]P_A + [d]Q_A)\rangle$$

be the isogeny effectively computed by Alice. We distinguish two cases.

Case 1: $\phi'_A \neq \psi_A$**.** Then $E_{AB} \neq E_B/\langle [2^a]R_a + [\alpha][2^a]S_a\rangle$ with overwhelming probability. In fact, if $E_{AB} = E_B/\langle [2^a]R_a + [\alpha][2^a]S_a\rangle$ with $\phi'_A \neq \psi_A$, then $\phi'_A \circ \widehat{\psi_A}$ is an endomorphism of E_{AB} of degree $2^{2a} \approx \sqrt{p}$. Since E_0 is a random supersingular curve, then the curve E_{AB} which is $2^a 2^b$ isogenous to E_0 can be assimilated to a random supersingular curve. Hence the probability that E_{AB} admits an endomorphism of degree $2^{2a} \approx \sqrt{p}$ is negligible.

Hence $R_{ab}, S_{ab} \notin E_B/\langle [2^a]R_a + [\alpha][2^a]S_a\rangle$. Therefore $[e_1]R_{ab} + [f_1]S_{ab} \neq \psi_A(R_a)$ and $[e_2]R_{ab}+[f_2]S_{ab} \neq \psi_A(S_a)$ since they are points on different curves.

Case 2: $\phi'_A = \psi_A$**.** Then Alice computes

$$\begin{aligned}
\psi_A(R_a) = \phi'_A(R_a) &= \phi'_A([x]\phi_B(P_2) + [y]\phi_B(Q_2)) \\
&= \phi'_B \circ \phi_A([x]P_2 + [y]Q_2) \\
&= \phi'_B([x]\phi_A(P_2) + [y]\phi_A(Q_2)) \\
&= \phi'_B\left([x]([e_1]R_A + [f_1]S_A) + [y]([e_2]R_A + [f_2]S_A)\right) \\
&= \phi'_B\left([xe_1 + ye_2]R_A + [xf_1 + yf_2]S_A\right) \\
&= [xe_1 + ye_2]\phi'_B(R_A) + [xf_1 + yf_2]\phi'_B(S_A)
\end{aligned}$$

and

$$\begin{aligned}
\psi_A(S_a) = \phi'_A(S_a) &= \phi'_A([z]\phi_B(P_2) + [t]\phi_B(Q_2)) \\
&= \phi'_A \circ \phi_B([z]P_2 + [t]Q_2) \\
&= \phi'_B([z]\phi_A(P_2) + [t]\phi_A(Q_2)) \\
&= \phi'_B\left([z]([e_1]R_A + [f_1]S_A) + [t]([e_2]R_A + [f_2]S_A)\right) \\
&= \phi'_B\left([ze_1 + te_2]R_A + [zf_1 + tf_2]S_A\right) \\
&= [ze_1 + te_2]\phi'_B(R_A) + [zf_1 + tf_2]\phi'_B(S_A)
\end{aligned}$$

On the other hand, Alice computes

$$[e_1]R_{ab} + [f_1]S_{ab} = [x'e_1 + z'f_1]\phi'_B(R_A) + [y'e_1 + t'f_1]\phi'_B(S_A)$$

and

$$[e_2]R_{ab} + [f_2]S_{ab} = [x'e_2 + z'f_2]\phi'_B(R_A) + [y'e_2 + t'f_2]\phi'_B(S_A)$$

The integers x, y, z, t, x', y', z' and t' need to satisfy

$$\begin{cases} \psi_A(R_a) = [e_1]R_{ab} + [f_1]S_{ab} \\ \psi_A(S_a) = [e_2]R_{ab} + [f_2]S_{ab} \end{cases}$$

i.e.

$$\begin{cases} [xe_1 + ye_2]\phi'_B(R_A) + [xf_1 + yf_2]\phi'_B(S_A) = [x'e_1 + z'f_1]\phi'_B(R_A) + [y'e_1 + t'f_1]\phi'_B(S_A) \\ [ze_1 + te_2]\phi'_B(R_A) + [zf_1 + tf_2]\phi'_B(S_A) = [x'e_2 + z'f_2]\phi'_B(R_A) + [y'e_2 + t'f_2]\phi'_B(S_A) \end{cases}$$

i.e.

$$\begin{cases} xe_1 + ye_2 = x'e_1 + z'f_1 \\ xf_1 + yf_2 = y'e_1 + t'f_1 \\ ze_1 + te_2 = x'e_2 + z'f_2 \\ zf_1 + tf_2 = y'e_2 + t'f_2 \end{cases} \quad \text{mod } 2^{2a}$$

i.e.

$$\begin{bmatrix} e_1 & e_2 & 0 & 0 \\ f_1 & f_2 & 0 & 0 \\ 0 & 0 & e_1 & e_2 \\ 0 & 0 & f_1 & f_2 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ t \end{bmatrix} = \begin{bmatrix} e_1 & 0 & f_1 & 0 \\ 0 & e_1 & 0 & f_1 \\ e_2 & 0 & f_2 & 0 \\ 0 & e_2 & 0 & f_2 \end{bmatrix} \begin{bmatrix} x' \\ y' \\ z' \\ t' \end{bmatrix} \quad \text{mod } 2^{2a} \tag{4}$$

From Remark 2, the knowledge of e_1, e_2, f_1 and f_2 is equivalent to the knowledge of Alice's private isogeny ϕ_A. Hence Bob does not have access neither to the matrix

$$M_1 = \begin{bmatrix} e_1 & e_2 & 0 & 0 \\ f_1 & f_2 & 0 & 0 \\ 0 & 0 & e_1 & e_2 \\ 0 & 0 & f_1 & f_2 \end{bmatrix} \in \mathcal{M}_2(\mathbb{Z}/2^{2a}\mathbb{Z}) \quad \text{nor} \quad M_2 = \begin{bmatrix} e_1 & 0 & f_1 & 0 \\ 0 & e_1 & 0 & f_1 \\ e_2 & 0 & f_2 & 0 \\ 0 & e_2 & 0 & f_2 \end{bmatrix} \in \mathcal{M}_2(\mathbb{Z}/2^{2a}\mathbb{Z}).$$

The solutions of Equation 4 that are independent of M_1 and M_2 satisfy

$$y = z = y' = z' = 0, \quad x = t = x' = t'.$$

Since $e_{2^{2a}}(R_a, S_a) = e_{2^{2a}}([a]\phi_B(P_2), [a]\phi_B(Q_2)) = e_{2^{2a}}(\phi_B(P_2), \phi_B(Q_2))^{a^2}$, then from the pairing equation $e_{2^{2a}}(R_a, S_a) = e_{2^{2a}}(P_2, Q_2)^{3^b}$, a needs to satisfy $a^2 = 1$, hence $a = \pm 1$.
We finally get $a = d = a' = d' = \pm 1$ and $b = c = b' = c' = 0$.

□

Remark 3. A formal proof of Theorem 1 can be obtained from that of Theorem 2 by setting $x = 1 = t$, $y = 0 = z$ or $x' = 1 = t'$, $y' = 0 = z'$ at the beginning of the proof depending on the points on which Bob decides to be partially point-malicious.

Remark 4. Bob can use the same key validation method to detect a malicious Alice. We set the isogeny degrees to powers of 2 and 3 just for simplicity. The key validation method generalises to any SIDH-like setup.

4 The HealSIDH (Healed SIDH) key exchange protocol

We now propose a variant of SIDH key exchange protocol which makes use of the GPST adaptive attack countermeasure we have just described. We first give the general idea behind the construction, then we concretely describe the key exchange and we finally discuss the underlying Diffie-Hellman-type hard problems.

4.1 An overview of HealSIDH

The idea behind HealSIDH is to incorporate the key validation mechanism described in Section 3 in the SIDH key exchange.

Set $p = 2^{2a}3^{2b}f - 1$ such that $2^a \approx 3^b$, $E_0[2^{2a}] = \langle P_2, Q_2\rangle$, $E_0[3^{2b}] = \langle P_3, Q_3\rangle$, $P_A = [2^a]P_2$, $Q_A = [2^a]Q_2$, $P_B = [3^b]P_3$ and $Q_B = [3^b]Q_3$. Alice's secret is an integer α sampled uniformly from $\mathbb{Z}/2^a\mathbb{Z}$ while Bob's secret is an integer β sampled uniformly from $\mathbb{Z}/3^b\mathbb{Z}$. Alice computes $\phi_A : E_0 \to E_A = E_0/\langle P_A + [\alpha]Q_A\rangle$ together with $\phi_A(P_2)$, $\phi_A(Q_2)$, $\phi_A(P_B)$ and $\phi_A(Q_B)$. She canonically generates the basis $\{R_A, S_A\}$ of $E_A[2^{2a}]$ and solves for e_1, f_1, e_2 f_2 such that $\phi_A(P_2) = [e_1]R_A + [f_1]S_A$ and $\phi_A(Q_2) = [e_2]R_A + [f_2]S_A$. Her public key is $(E_A, \phi_A(P_B), \phi_A(Q_B))$ and her secret key is $(\alpha, e_1, f_1, e_2, f_2)$. Bob does the same to obtain a public key $(E_B, \phi_B(P_A), \phi_B(Q_A))$ and a secret key $(\beta, g_1, h_1, g_2, h_2)$.

If Bob wishes to establish a shared secret with Alice, he retrieves Alice's public key (E_A, R_b, S_b), computes $\phi'_B : E_A \to E_{AB} = E_A/\langle [3^b]R_b + [\beta][3^b]S_b\rangle$ together with $\phi'_B(R_A)$, $\phi'_B(S_A)$, $\phi'_B(R_b)$ and $\phi'_B(S_b)$. The yet to be confirmed shared secret is the j-invariant j_{AB} of E_{AB}. He sends $(\phi'_B(R_A), \phi'_B(S_A))$ to Alice.

Upon receiving $(\phi'_B(R_A), \phi'_B(S_A))$, Alice retrieves Bob's public key (E_B, R_a, S_a). She computes $\phi'_A : E_B \to E_{BA} = E_B/\langle [2^a]R_a + [\alpha][2^a]S_a\rangle$ together with $\phi'_A(R_B)$, $\phi'_A(S_B)$, $\phi'_A(R_a)$ and $\phi'_A(S_a)$. She then computes R_{ba} and $\widehat{\phi'_A}(\phi'_B(S_A))$.
If $e_{2^{2a}}(R_a, S_a) \neq e_{2^{2a}}(P_2, Q_2)^{3^b}$ or $[e_1]\phi'_B(R_A) + [f_1]\phi'_B(S_A) \neq \phi'_A(R_a)$ or $[e_2]\phi'_B(R_A) + [f_2]\phi'_B(S_A) \neq \phi'_A(S_a)$, Alice aborts. Otherwise, she sends $\phi'_A(R_B)$ and $\phi'_A(S_B)$ to Bob and keeps the j-invariant j_{BA} of E_{BA} as the shared secret.

Upon receiving $\phi'_A(R_B)$ and $\phi'_A(S_B)$, Bob does the key validation check. If $e_{3^{2b}}(R_b, S_b) \neq e_{3^{2b}}(P_3, Q_3)^{2^a}$ or $[g_1]\phi'_A(R_B) + [h_1]\phi'_A(S_B) \neq \phi'_B(R_b)$ or $[g_2]\phi'_A(R_B) + [h_2]\phi'_A(S_B) \neq \phi'_B(S_b)$, Bob aborts . If not he successfully takes j_{AB} as the shared secret.

Practically, if Bob reveals the points $\phi'_B(R_A)$ and $\phi'_B(S_A)$, or Alice reveals $\phi'_A(R_B)$ and $\phi'_A(S_B)$, then an adversary can recover the curve E_{AB} since for $P \in E_{AB}$, the Montgomery coefficient $A_{E_{AB}}$ of E_{AB} satisfies $A_{E_{AB}} = \frac{y(P)^2 - x(P)^3 - x(P)}{x(P)^2}$. We avoid this by exploiting the ideas used in SIKE [19] for key compression: represent a point $P \in E[N]$ by its coordinates in a basis of $E[N]$ which can be canonically computed.

4.2 HealSIDH Key Exchange

Instead of revealing the points $\phi'_B(R_A)$ and $\phi'_B(S_A)$, Bob canonically generates a basis $\{R_{AB}, S_{AB}\}$ of $E_{AB}[2^{2a}]$ and computes $e_3, f_3, e_4, f_4 \in \mathbb{Z}_{2^{2a}}$ such that

$$\phi'_B(R_A) = [e_3]R_{AB} + [f_3]S_{AB} \text{ and } \phi'_B(S_A) = [e_4]R_{AB} + [f_4]S_{AB}.$$

Similarly, Alice canonically generates a basis $\{R_{BA}, S_{BA}\}$ of $E_{BA}[3^{2b}]$ and computes $g_3, h_3, g_4, h_4 \in \mathbb{Z}_{3^{2b}}$ such that

$$\phi'_A(R_B) = [g_3]R_{BA} + [h_3]S_{BA} \text{ and } \phi'_A(S_B) = [g_4]R_{BA} + [h_4]S_{BA}.$$

Concretely, the HealSIDH Key Exchange is entirely described in Figure 2.

Lemma 1. *HealSIDH is correct.*

Proof. Follows from the correctness of SIDH and Theorem 2. □

Remark 5. Two parties Alice and Bob need to run the key validation only once, during their first communication. In the subsequent communications between the two parties there is no need to revalidate the keys.

4.3 Security of HealSIDH

We present the Computational Diffie-Hellman-type problem underlying the security of HealSIDH. We argue that the Decisional variant of this problem is not hard.

Problem 4 (HealSIDH-CDHP). Let $p = 2^{2a}2^{2b}f - 1$ and E_0 a supersingular curve defined over $\mathbb{F}_{p^2}$ with unknown endomorphism ring. Let $E_0[2^{2a}] = \langle P_2, Q_2\rangle$, $E_0[3^{2b}] = \langle P_3, Q_3\rangle$, $P_A = [2^a]P_2$, $Q_A = [2^a]Q_2$, $P_B = [3^b]P_3$, $Q_B = [3^b]Q_3$. Let $\phi_A : E_0 \to E_A$, $\phi_B : E_0 \to E_B$, $\phi'_A : E_B \to E_{BA}$ and $\phi'_B : E_A \to E_{AB}$ be secret isogenies as described in SIDH-type schemes. Let $E_A[2^{2a}] = \langle R_A, S_A\rangle$, $E_B[3^{2b}] = \langle R_B, S_B\rangle$, $E_{AB}[2^{2a}] = \langle R_{AB}, S_{AB}\rangle$, $E_{AB}[3^{2b}] = \langle R_{BA}, S_{BA}\rangle$. Let $e_3, f_3, e_4, f_4 \in \mathbb{Z}_{2^{2a}}$ and $g_3, h_3, g_4, h_4 \in \mathbb{Z}_{3^{2b}}$ such that $\phi'_A(R_B) = [g_3]R_{BA} + [h_3]S_{BA}$, $\phi'_A(S_B) = [g_4]R_{BA} + [h_4]S_{BA}$, $\phi'_B(R_A) = [e_3]R_{AB} + [f_3]S_{AB}$ and $\phi'_B(S_A) = [e_4]R_{AB} + [f_4]S_{AB}$.

Given E_0, P_2, Q_2, P_3, Q_3, E_A, $\phi_A(P_2)$, $\phi_A(Q_3)$, R_A, S_A, E_B, $\phi_B(P_2)$, $\phi_B(Q_2)$, R_B, S_B, e_3, f_3, e_4, f_4, g_3, h_3, g_4, h_4, compute E_{AB}.

The main differences between Problem 4 and Problem 1 are as follows:

- the action of the secret isogeny ϕ_A (resp. ϕ_B) of degree 2^a (resp. 3^b) on $E_0[3^{2b}]$ (resp. $E_0[2^{2a}]$) is revealed;
- in addition to image points through ϕ_A as in SIDH, the coordinates of some image points through ϕ'_A (resp. ϕ'_B) in a canonical basis are revealed.

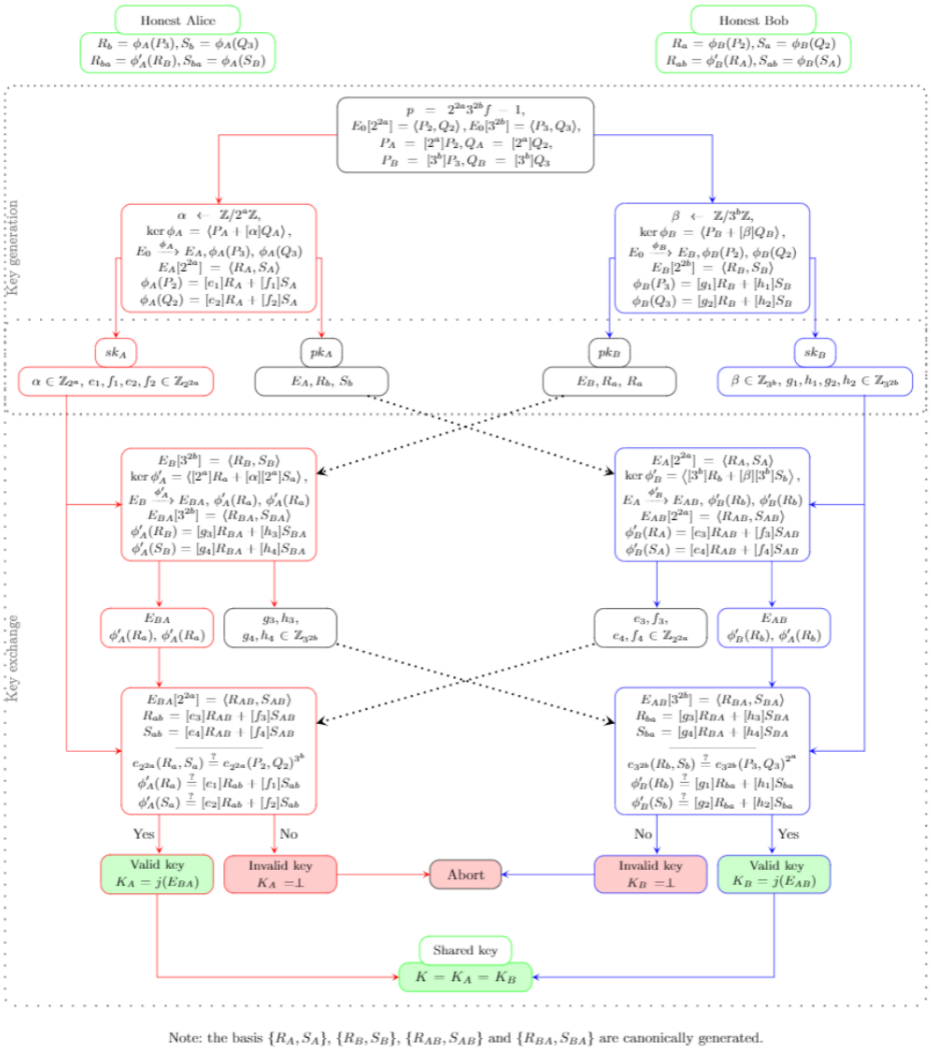

Fig. 2: HealSIDH interactive key exchange. E_0 is a random supersingular curve.

With respect to the first point, we reveal the action of isogenies of degree $A \approx p^{1/4}$ on a B-torsion subgroup where $B \approx p^{1/2}$. Since the endomorphism ring of the curve E_0 is unknown, then HealSIDH is protected against improved torsion attacks [10].

With respect to the second point, the coordinates g_3, h_3, g_4, h_4 of $\phi'_A(R_B)$ and $\phi'_A(S_B)$ in a canonical basis of $E_{BA}[3^{2b}]$, and the coordinates e_3, f_3, e_4, f_4 of $\phi'_B(R_A)$ and $\phi'_B(S_A)$ in a canonical basis of $E_{BA}[2^{2a}]$ are revealed. We don't see how this could affect the hardness of Problem 4.

Nevertheless, revealing these coordinates implies that the decisional version of Problem 4 is not hard. In fact, suppose that you are given a random supersingular elliptic curve E and you wish to determine if $E = E_{BA}$ or $E \neq E_{BA}$. Then you can generate the canonical bases $E[3^{2b}] = \langle R_{BA}, S_{BA}\rangle$ and $E[2^{2a}] = \langle R_{AB}, S_{AB}\rangle$, perform the pairing checks

$$e_{2^{2a}}\left([e_3]R_{AB} + [f_3]S_{AB}, [e_4]R_{AB} + [f_4]S_{AB}\right) \stackrel{?}{=} e_{2^{2a}}(R_A, S_A)^{3^b}$$

and

$$e_{3^{2b}}\left([g_3]R_{BA} + [h_3]S_{BA}, [g_4]R_{BA} + [h_4]S_{BA}\right) \stackrel{?}{=} e_{3^{2b}}(R_B, S_B)^{2^a}.$$

If $E = E_{AB}$, then these checks would be successful. If $E \neq E_{AB}$, then these checks will fail with overwhelming probability since the points $[e_3]R_{AB}+[f_3]S_{AB}$, $[e_4]R_{AB}+[f_4]S_{AB}$, $[g_3]R_{BA}+[h_3]S_{BA}$ and $[g_4]R_{BA}+[h_4]S_{BA}$ would be random points of E of order 2^{2a}, 2^{2a}, 3^{2b} and 3^{2b} respectively; hence likely would not satisfy the pairing equalities.

5 SHealS: a public key encryption scheme

Even though the DDH-type problem for HealSIDH is not hard, we still use HealSIDH to design a secure public key encryption scheme, which we call SHealS. We first give an overview of our construction, then we fully describe and analyze it.

5.1 An overview of SHealS

Our aim is to derive a PKE scheme from HealSIDH.

A canonical way to design a PKE scheme from HealSIDH is to proceed as follows. Consider the HealSIDH setting. Alice generates her key pair $(\mathsf{sk}_A, \mathsf{pk}_A)$ where $\mathsf{sk}_A = (\alpha, e_1, f_1, e_2, f_2)$ and $\mathsf{pk}_A = (E_A, R_b, S_b)$. In order to encrypt a plaintext m of binary length n, Bob randomly samples $\beta \in \mathbb{Z}/3^b\mathbb{Z}$, computes $\mathsf{c}_0 = (E_B, R_a, S_a, e_3||f_3||e_4||f_4)$ and $\mathsf{c}_1 = H(j_{AB}) \oplus \mathsf{m}$ where $H : \mathbb{F}_{p^2} \to \{0,1\}^n$ is a cryptographic hash function. The ciphertext is $\mathsf{c} = (\mathsf{c}_0, \mathsf{c}_1)$. Decryption consists in completing the underlying HealSIDH key exchange using sk_A and c_0. If the key exchange is successful, recover $\mathsf{m} = \mathsf{c}_1 \oplus H(j_{BA})$ using the shared secret E_{BA}, else $\mathsf{m} = \perp$.

As shown in the following lemma, the resulting PKE scheme is not IND-CCA secure.

Lemma 2. *Let $\mathsf{m} \in \{0,1\}^n$ be a plaintext and let $k \geq 1$ be an integer such that the k^{th} bit of m (the coefficient of 2^{k-1} in the 2-adic expansion of m) is 0. If $\mathsf{c} = (\mathsf{c}_0, \mathsf{c}_1)$ is a ciphertext for m, then $\mathsf{c}' = (\mathsf{c}_0, \mathsf{c}_1 \oplus 2^{k-1})$ is a ciphertext for $\mathsf{m} + 2^{k-1}$.*

Proof. Since the k^{th} bit of m is 0, then $\mathsf{m} + 2^{k-1} = \mathsf{m} \oplus 2^{k-1}$. Hence

$$\mathsf{c}_1 \oplus 2^{k-1} = \mathsf{m} \oplus H(j_{AB}) \oplus 2^{k-1} = (\mathsf{m} \oplus 2^{k-1}) \oplus H(j_{AB}) = (\mathsf{m} + 2^{k-1}) \oplus H(j_{AB}).$$

Therefore $\mathsf{c}' = (\mathsf{c}_0, \mathsf{c}_1 \oplus 2^{k-1})$ is a ciphertext for $\mathsf{m} + 2^{k-1}$. □

This IND-CCA attack applies to all PKE schemes in which the ciphertext is of the form $(\mathsf{c}_0, H(s) \oplus \mathsf{m})$ where s and c_0 are independent of m. We choose to use points to encrypt the plaintext, as in SiGamal [23] and SimS [14].

5.2 SHealS Public Key Encryption scheme

The plaintext space is changed to $\mathcal{M} = \mathbb{Z}_{2^{2a}}^{\times}$, the set invertible elements in the ring of integers modulo 2^{2a}. The ciphertext of a given plaintext $\mathsf{m} \in \mathcal{M}$ is $\mathsf{c} = (\mathsf{c}_0, \mathsf{c}_1)$ where $\mathsf{c}_0 = (E_B, R_a, S_a)$, $\mathsf{c}_1 = H(j_{AB}) \oplus (\mathsf{m}e_3 || \mathsf{m}f_3 || \mathsf{m}e_4 || \mathsf{m}f_4)$ and $H : \mathbb{F}_{p^2} \to \{0,1\}^{8a}$ is a cryptographic hash function.

Note that scaling e_3, f_3, e_4 and f_4 by m is equivalent to scaling the points $[e_3]R_{AB} + [f_3]S_{AB}$ and $[e_4]R_{AB} + [f_4]S_{AB}$ by $[\mathsf{m}]$. This enables Alice to recover m by solving a discrete logarithm instance in a group of order 2^{2a}.

Concretely, Figure 3 entirely describes SHealS PKE.

Lemma 3. *SHealS PKE is correct.*

Proof. Follows from the correctness of HealSIDH.

Remark 6. In SHealS, since there is no key disclosure, Bob can reuse his encryption key β to encrypt other plaintexts. Moreover, since the 3^{2b} torsion points are readily available, he can use the same β as a static private key.

5.3 Security analysis

We prove that SHealS is IND-CPA secure relying on Assumption 1. Next we discuss the IND-CCA security of SHealS. We conjecture that SHealS is IND-CCA secure and provide arguments to support our conjecture.

Assumption 1 *Let E_0, P_2, Q_2, P_A, Q_A,P_3, Q_3, P_B, Q_B, E_A, R_A, S_A, $\phi_A(P_3)$, $\phi_A(Q_3)$, E_B, $\phi_B(P_2)$, $\phi_B(Q_2)$ the public parameters and keys of an HealSIDH instance. Set $E_{AB}[2^{2a}] = \langle R_{AB}, S_{AB} \rangle$ where the basis $\{R_{AB}, S_{AB}\}$ is canonically generated, let $\mathcal{B}_0 = \{\phi'_B(R_A), \phi'_B(S_A)\}$ and let $\mathcal{B}_1 = \{R, S\}$ be a uniformly random basis of $E_{AB}[2^{2a}]$ such that $e_{2^{2a}}(R,S) = e_{2^{2a}}(R_A, S_A)^{3^b}$. Set $\phi'_B(R_A) = [e_{03}]R_{AB} + [f_{03}]S_{AB}$, $\phi'_B(S_A) = [e_{04}]R_{AB} + [f_{04}]S_{AB}$, $R = [e_{13}]R_{AB} + [f_{13}]S_{AB}$ and $S = [e_{14}]R_{AB} + [f_{14}]S_{AB}$. For any PPT algorithm $\mathcal{A}$,*

$$Pr\left[b = b^* \;\middle|\; \begin{array}{l} b \xleftarrow{\$} \{0,1\}, \\ b^* \leftarrow \mathcal{A}\begin{pmatrix} E_A, \phi_A(P_3), \phi_A(Q_3), E_B, \phi_B(P_2), \\ \phi_B(Q_2), E_{AB}, e_{b3} || f_{b3} || e_{b4} || f_{b4} \end{pmatrix} \end{array} \right] = \frac{1}{2} + \mathsf{negl}(\lambda).$$

Theorem 3. *If Assumption 1 holds, then SHealS is IND-CPA secure.*

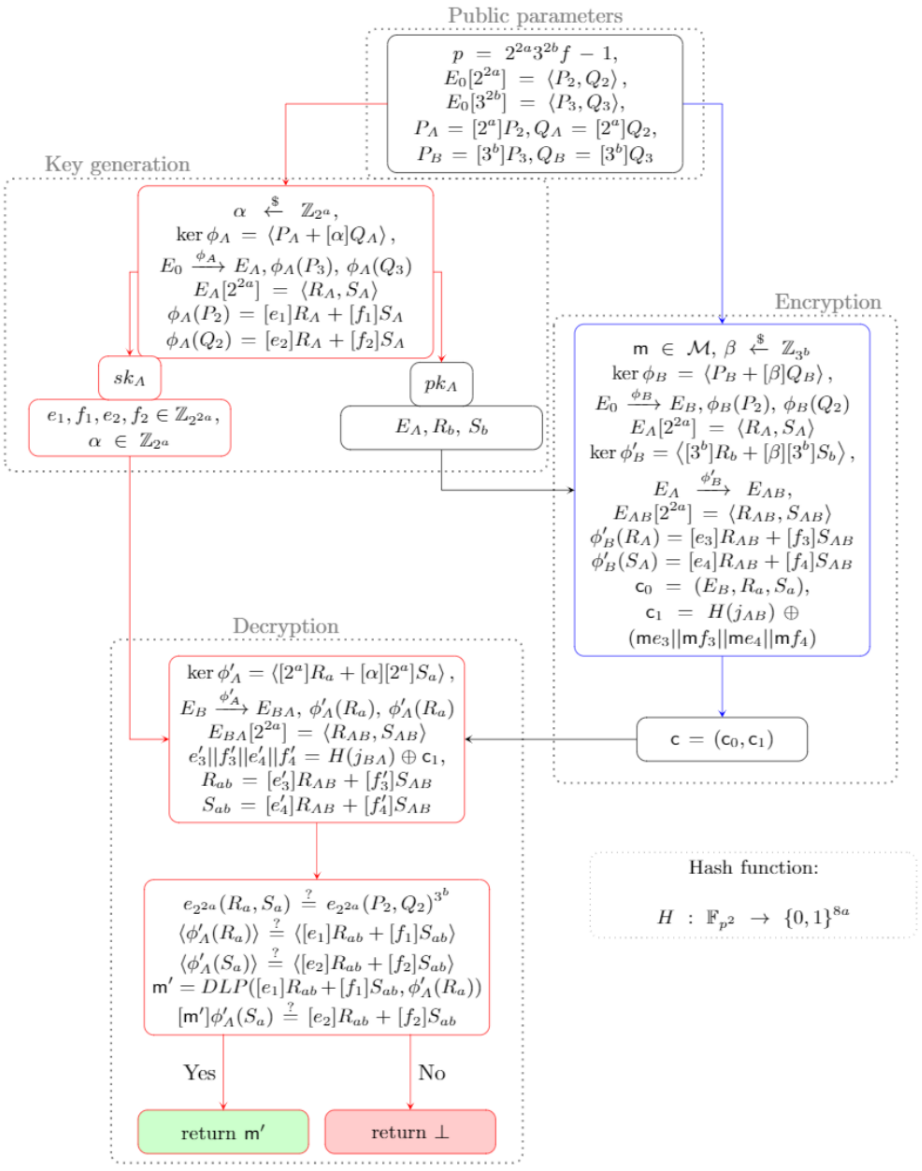

Fig. 3: SHealS PKE. E_0 is a supersingular curve with unknown endomorphism ring.

Proof. Analogous to the proof of [14, Theorem 3]. □

Concretely, Assumption 1 states that given E_A, $\phi_A(P_3)$, $\phi_A(Q_3)$, E_B, $\phi_B(P_2)$, $\phi_B(Q_2)$, E_{AB}, it is difficult to distinguish the images points $\phi'_B(R_A), \phi'_B(S_A)$ of

a basis $\{R_A, S_A\}$ of $E_A[2^{2a}]$ through ϕ'_B and a uniformly random basis $\{R, S\}$ of $E_{AB}[2^{2a}]$ such that $e_{2^{2a}}(R,S) = e_{2^{2a}}(R_A, S_A)^{3^b}$.

Concerning the IND-CCA security of SHealS, one may be tempted to use a knowledge of exponnent type as Fouotsa and Petit did to prove the IND-CCA security of SimS [14]. But this type of assumption does not hold for SIDH type schemes. In fact, one can not see SIDH as an analog to the classic Diffie-Hellman as it is the case in CSIDH. In CSIDH, the secret isogeny can have any degree in a well chosen key space. But in SIDH, the degree of the secret isogeny is fixed. This eliminates the idea of assimilating the secret isogenies in SIDH to "exponents".

We have not been able to come up with a succinct proof of IND-CCA security for SHealS, but we argue that SHealS is not vulnerable to any known attack on SIDH type schemes since we have countered the GPST adaptive attack [17] and possible variants of it, and the improved torsion points attacks [26,10]. Note that we do not take side channel attacks into consideration in this analysis. We hence state the following conjecture and leave it's proof or its invalidation for future work.

Conjecture 4 *SHealS is IND-CCA secure.*

6 Concrete instantiations and comparisons: HealSIDH vs k-SIDH; SHealS vs SIKE

6.1 Concrete instantiation

We performed a basic Sagemath [29] proof-of-concept implementation of our key validation method, HealSIDH and SHealS. We use the prime $p_{870} = 2^{432}3^{274}10 - 1$ where $a = 216$ and $b = 137$ as in SIKEp434 [19, §1.6]. Hence we expect SHealSp870 and SIKEp434 on one hand, HealSIDHp870 and k-SIDHp434 on the other hand, to provide the same security level.

The proof-of-concept implementation of SHealS is very basic and unoptimized, hence it cannot serve as a reference when comparing SHealS and SIKE in terms of efficiency. In the following paragraph, we do a high level comparison between SHealS and SIKE. We argue that the efficiency of an optimized implementation of SHealS is comparable to that of SIKE (considering instances providing the same security level).

6.2 SHealS vs SIKE

We provide a high level comparison between SHealS and SIKE and argue that SHealS's efficiency is close to that of SIKE. In what follows, we suppose that in both SIKE and SHealS, an SIDH-type public key (E, P, Q) is represented by (x_P, x_Q, x_{P-Q}) as specified in [19]. Let λ be a security parameter, and let p_h and p_s respectively be the HealSIDH (or SHealS) prime and the SIKE prime providing λ bits of security. It follows that $\lceil \log p_s \rceil \approx 4\lambda$ and $\lceil \log p_h \rceil \approx 8\lambda$.

Design. At the design level, in SHealS, the encryption public key is validated through a "direct" key validation mechanism while in SIKE, the validation is done through re-encryption. For this reason, the number of isogenies computed in SIKE ($\mathsf{KeyGeneration} + \mathsf{Encapsulation} + \mathsf{Decapsulation}$) is 5 while only 4 isogenies are computed in SHealS ($\mathsf{KeyGeneration} + \mathsf{Encryption} + \mathsf{Decryption}$). Nevertheless, all the 4 isogenies in SHealS are evaluated on torsion points as well, while only 3 of the 5 isogenies in SIKE are evaluated on torsion points. In SHealS, a trusted party is needed for the generating the starting curve E_0.

Security. SHealS's IND-CCA security is conjectured while that of SIKE is inheritated from a variant Fujisaki-Okamoto transform [18].

Keys sizes. In SIKE and SHealS, the secret key is α and $(\alpha, e_1, f_1, e_2, f_2)$ respectively. Since e_1, f_1, e_2, f_2 lie in $\mathbb{Z}/2^{2a}\mathbb{Z}$, then their bitsize is twice that of $\alpha \in \mathbb{Z}_{2^a}$. Hence the secret key of HealSIDH is 9 times larger compared to that of SIKE. The public key in SIKE and SHealS are both of the form (x_P, x_Q, x_{P-Q}). Hence in SIKE the public key has roughly $3(2\lceil \log p_s \rceil) = 6\lceil \log p_s \rceil \approx 24\lambda$ bits while in SHealS it has roughly $3(2\lceil \log p_h \rceil) = 6\lceil \log p_h \rceil \approx 48\lambda$ bits. Therefore, the size of the public key in SHealS is roughly twice that of the public key in SIKE.
For the ciphertext, the bitsize of c_0 in SHealS is twice that of c_0 in SIKE, while the bit size of c_1 in ShealS is $8a = 16\lambda$, opposed to $n \in \{128, 192, 256\}$ in SIKE. It follows that the size of SHealS ciphertexts is about 2.45 times that of SIKE ciphertexts.

Efficiency. As mentioned before, only 4 isogenies are computed in SHealS while 5 isogenies are computed in SIKE. Meanwhile, the prime used in SHealS is twice as large as SIKE prime. And, in SHealS, the isogenies $\phi'_A : E_B \to E_{BA}$ and $\phi'_B : E_A \to E_{AB}$ are evaluated on two torsion points each, which is not the case in SIKE. Without an advanced implementation of SHealS, it is difficult to provide a precise efficiency comparison between both schemes.

We summarize the comparison in Table 1. Let λ be a desired security level.

6.3 HealSIDH vs k-SIDH

To the best of our knowledge, the only existing post-quantum key exchange schemes enabling static-static key setting prior to this work[4] were CSIDH [4], k-SIDH [1] and its variant by Jao and Urbanik [30]. As highlighted in Section 2.6, Basso et al. [2] showed that k-SIDH is preferable to the later variant from an efficiency vs security point of view. We provide a high level comparison between HealSIDH and k-SIDH since both are countermeasures to the GPST adaptive attacks.

Design. At the design level, HealSIDH comes with an incorporated key validation method, while k-SIDH mitigates the GPST adaptive attacks by running many parallel SIDH intances. This implies that more than k^2 isogenies are computed in

[4] While this work was under submission, a proof of isogeny knowledge [12] was published online. We will provide a concrete comparison with this construction in later versions of this paper that we will make available on the IACR eprint database.

	SIKE	SHealS
Field characteristic size	$\approx 4\lambda$	$\approx 8\lambda$
Private key size	$\approx 2\lambda$	$\approx 18\lambda$
Public key size	$\approx 24\lambda$	$\approx 48\lambda$
Ciphertext size	$\approx 26\lambda$	$\approx 64\lambda$
KeyGen (isog. comp.)	1	1
Encaps/Encrypt (isog. comp.)	2	2
Decaps/Decrypt (isog. comp.)	2	1
Adaptive attacks	No	No (conjecture)
Key disclosure	Yes	No
Encryption key reuse	No	Yes
Key validation method used	re-encryption	Key val. method presented in § 3

Table 1: High level comparison between SHealSIDH and SIKE.

k-SIDH (full execution of the key exchange) while only 4 isogenies are computed in HealSIDH. Nevertheless, There are two rounds in HealSIDH, as opposed to one round in k-SIDH. Note that the starting curve in HealSIDH is generated by a trusted party, which is not the case in k-SIDH.

Security. Security wise, HealSIDH is not vulnerable to the GPST adaptive attacks since it incorporates a countermeasure. In k-SIDH, one does not eliminate the attack completely, but one increases its cost in such a way that it becomes exponential in k.

Keys sizes. From the comparison made in Section 6.2, the secret key in HealSIDH has roughly 18λ bits. In k-SIDH, the size of the secret key is k times that of a SIKE secret key, hence $2k\lambda$. The public keys in HealSIDH have roughly 48λ bits while in k-SIDH they have about $24k\lambda$ bits.

Efficiency. As mentioned before, only 4 isogenies are computed in HealSIDH. In k-SIDH, roughly $2k^2 + 2k$ isogenies are computed. Even though the HealSIDH prime size is twice that of the k-SIDH prime, k-SIDH is still an order of magnitude less efficient compared to HealSIDH because of the relatively large number of isogenies computed.

Table 2 provides a high level comparison between HealSIDH and k-SIDH. We refer to [1] for more details on k-SIDH.

7 HealS (Healed SIKE): improving the efficiency of SHealS

From the comparison in Section 6.2, one concludes that the prime size, the key and ciphertext sizes in SHealS are at least twice that in SIKE. In this section, our aim is to improve on this prime, key and ciphertext sizes.

	HealSIDH	k-SIDH
Field characteristic size	$\approx 8\lambda$	$\approx 4\lambda$
Private key size	$\approx 18\lambda$	$\approx 2k\lambda$
Public key size	$\approx 48\lambda$	$\approx 24k\lambda$
KeyGen	1	k
key exchange	2	$2k^2$
Adaptive attacks	No	exp. in k
Static-static key	yes	yes
NIKE	No	yes

Table 2: High level comparison between HealSIDH and k-SIDH ($46 \leq k$).

7.1 HealS Public Key Encryption

Having a closer look at ShealS, one notices that since Bob does not run a key validation on Alice's public key in the PKE encryption scheme, then it is not a requisite to have the 3^{2b}-torsion points defined over $\mathbb{F}_{p^2}$. Hence when the parameters are chosen for a PKE scheme purpose only, the prime p can be relaxed to $p = 2^{2a}3^b f - 1$ where $2^a \approx 3^b$ and f is a small cofactor. Most of the scheme remains unchanged. Concretely, HealS is SHealS with a prime of the form $p = 2^{2a}3^b f - 1$.

While the base prime change when going from SHealS to HealS comes with considerable speed-up and considerable improvement on key and ciphertext sizes (see Section 7.2), one should notice that Bob can no more use his encryption key as secret key when receiving encrypted messages. In fact, in order to encrypt a plaintext for Bob, one needs to compute the images of torsion points of order 3^{2b}. For HealS primes, these torsion points are defined over large extensions since $p = 2^{2a}3^b f - 1$. Nevertheless, Bob can reuse the same encryption key β to encrypt other messages to other parties or the same party, only he can not use it as decryption key. This technical difference motivated us to rename the instance HealS instead of keeping the name SHealS. Appendix A provides more details about the KeyGeneration, Encryption and Decryption algorithms in HealS.

7.2 Concrete instantiation and comparison with SIKE

We instatiate HealS with the prime $p_{650} = 2^{432}3^{137} - 1$ where $a = 216$ and $b = 137$ as in SIKEp434 [19, §1.6]. Hence HealSp650 and SIKEp434 are expected to provide the same security level.

We summarise a high level comparison between HealS and SIKE in Table 3. We also include SHealS in this table to highlight the advantages of HealS when compared to SHealS.

Table 4 compares the key and ciphertext sizes of our PKE with some NIST finalists KEMs. We notice that the key sizes in HealS are more compact compared to these finalists. The ciphertext size in HealS is close to that of Kyber, NTRU

	SIKE	SHealS	HealS
Field characteristic size	$\approx 4\lambda$	$\approx 8\lambda$	$\approx 6\lambda$
Private key size	$\approx 2\lambda$	$\approx 18\lambda$	$\approx 18\lambda$
Public key size	$\approx 24\lambda$	$\approx 48\lambda$	$\approx 36\lambda$
Ciphertext size	$\approx 26\lambda$	$\approx 64\lambda$	$\approx 48\lambda$
KeyGen (isog. comp.)	1	1	1
Encaps/Encrypt (isog. comp.)	2	2	2
Decaps/Decrypt (isog. comp.)	2	1	1
Adaptive attacks	No	No (conj.)	No (conj.)
Key disclosure	Yes	No	No
Encryption key reuse	No	Yes	Yes
Key validation method used	re-encryption	Key val. method presented in § 3	

Table 3: High level comparison between HealS, SHealS and SIKE.

and Saber, while being considerably larger compared to that of Classic McEliece.

	HealS	Kyber	NTRU	Classic McEliece	Saber
sk	288	1632	935	6492	1568
pk	576	800	699	261120	672
c	768	768	699	128	736

Table 4: Key and ciphertext sizes comparison between HealS and the four NIST finalists KEMs Kyber, NTRU, Classic McEliece and Saber, for 128 bits of security (NIST level 1).

8 Conclusion

In this paper, we introduced an efficient countermeasure to the GPST adaptive attack which does not require key disclosure nor re-encryption. Next, we used this countermeasure to design an efficient static-static key interactive exchange scheme: HealSIDH. HealSIDH is not vulnerable to the GPST adaptive attacks. We derive an IND-CPA secure PKE scheme with conjectured IND-CCA security SHealS from HealSIDH. The full execution of SHealS contains only 4 isogeny computations while that of SIKE contains 5 isogeny computations. For this reason, even though SHealS uses larger parameters and has larger keys, we expect its efficiency to be comparable to that of SIKE. In order to optimize the efficiency, keys and ciphertexts sizes, we suggest HealS, a variant of SHealS using a smaller prime. The main difference between SHealS and HealS is that in SHealS,

a party can use his private key as encryption key when encrypting ciphertexts for other parties.

Moreover, we provided a high level comparison between HealSIDH and k-SIDH on one hand, and between SHealS, HealS and SIKE on the other hand. HealSIDH is an order of magnitude more efficient compared to k-SIDH and the keys in k-SIDH are about k times bigger compared to those of HealSIDH. The advantages of SHealS and HealS over SIKE are

- no encryption key disclosure to the recipient during encryption;
- incorporated key validation method (no re-encryption during decryption);
- encryption key reuse.

In order to evaluate the concrete efficiency of the schemes constructed in this paper, an advanced implementation of SHealS and HealS is needed. We leave this task to follow-up work. We believe the design of SHealS leaves room for considerable optimisations. These may come from the implementation, from variants of the key validation method or from redesigning the schemes.

Furthermore, there are possibly existing isogeny-based schemes that would benefit from our key validation method. Also the key validation may enables the design of new isogeny-based primitives. We also leave such an investigation for future work.

Acknowledgements. We would like to express our sincere gratitude to the anonymous reviewers for their helpful comments and suggestions. Christophe Petit was supported by EPSRC grant EP/S01361X/1.

References

1. Reza Azarderakhsh, David Jao, and Christopher Leonardi. Post-quantum static-static key agreement using multiple protocol instances. In *International Conference on Selected Areas in Cryptography*, pages 45–63. Springer, 2017.
2. Andrea Basso, Péter Kutas, Simon-Philipp Merz, Christophe Petit, and Charlotte Weitkämper. On adaptive attacks against jao-urbanik's isogeny-based protocol. In Abderrahmane Nitaj and Amr Youssef, editors, *Progress in Cryptology - AFRICACRYPT 2020*, pages 195–213, Cham, 2020. Springer International Publishing.
3. Xavier Bonnetain and André Schrottenloher. Quantum Security Analysis of CSIDH. *Advances in Cryptology – EUROCRYPT 2020*, 12106:493 – 522, 2020.
4. Wouter Castryck, Tanja Lange, Chloe Martindale, Lorenz Panny, and Joost Renes. CSIDH: an efficient post-quantum commutative group action. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 395–427. Springer, 2018.
5. Denis X Charles, Kristin E Lauter, and Eyal Z Goren. Cryptographic hash functions from expander graphs. *Journal of Cryptology*, 22(1):93–113, 2009.
6. Andrew Childs, David Jao, and Vladimir Soukharev. Constructing elliptic curve isogenies in quantum subexponential time. *Journal of Mathematical Cryptology*, 8(1):1–29, 2014.

7. Jorge Chávez-Saab, Jesús-Javier Chi-Domínguez, Samuel Jaques, and Francisco Rodríguez-Henríquez. The SQALE of CSIDH: Square-root vélu Quantum-resistant isogeny Action with Low Exponents. Cryptology ePrint Archive, Report 2020/1520, 2020. https://eprint.iacr.org/2020/1520.
8. Jean-Marc Couveignes. Hard homogeneous spaces. Cryptology ePrint Archive, Report 2006/291, 2006. https://eprint.iacr.org/2006/291.
9. Luca De Feo, Simon Masson, Christophe Petit, and Antonio Sanso. Verifiable delay functions from supersingular isogenies and pairings. In Steven D. Galbraith and Shiho Moriai, editors, *Advances in Cryptology – ASIACRYPT 2019*, pages 248–277, Cham, 2019. Springer International Publishing.
10. Victoria de Quehen, Péter Kutas, Chris Leonardi, Chloe Martindale, Lorenz Panny, Christophe Petit, and Katherine E. Stange. Improved torsion point attacks on SIDH variants. Cryptology ePrint Archive, Report 2020/633, 2020. https://eprint.iacr.org/2020/633.
11. Samuel Dobson, Steven D. Galbraith, Jason LeGrow, Yan Bo Ti, and Lukas Zobernig. An adaptive attack on 2-sidh. *International Journal of Computer Mathematics: Computer Systems Theory*, 5(4):282–299, 2020.
12. Luca De Feo, Samuel Dobson, Steven D. Galbraith, and Lukas Zobernig. Sidh proof of knowledge. Cryptology ePrint Archive, Report 2021/1023, 2021. https://ia.cr/2021/1023.
13. Luca De Feo, David Jao, and Jérôme Plût. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies, 2014. Pagesn 209-247.
14. Tako Boris Fouotsa and Christophe Petit. Sims: A simplification of sigamal. In Jung Hee Cheon and Jean-Pierre Tillich, editors, *Post-Quantum Cryptography*, pages 277–295, Cham, 2021. Springer International Publishing.
15. Eiichiro Fujisaki and Tatsuaki Okamoto. Secure integration of asymmetric and symmetric encryption. In Michael J. Wiener, editor, Advances in Cryptology-CRYPTO'99, volume 1666 of Lecture Notes in Computer Science, pages 537-554, Santa Barbara, CA, USA, August 15-19, 1999. Springer, Heidelberg, Germany.
16. Steven D. Galbraith. *Mathematics of Public Key Cryptography*. Cambridge University Press, 2012.
17. Steven D Galbraith, Christophe Petit, Barak Shani, and Yan Bo Ti. On the security of supersingular isogeny cryptosystems. In *Advances in Cryptology – ASIACRYPT 2016*, pages 63–91. Springer, 2016.
18. Dennis Hofheinz, Kathrin Hövelmanns, and Eike Kiltz. A modular analysis of the Fujisaki-Okamoto transformation. In *Theory of Cryptography Conference*, pages 341–371. Springer, 2017.
19. David Jao, Reza Azarderakhsh, Matthew Campagna, Craig Costello, Luca De Feo, Basil Hess, Aaron Hutchinson, Amir Jalali, Koray Karabina, Brian Koziel, Brian LaMacchia, Patrick Longa, Michael Naehrig, Geovandro Pereira, Joost Renes, Vladimir Soukharev, and David Urbanik. Supersingular Isogeny Key Encapsulation, October 1, 2020. https://sike.org/files/SIDH-spec.pdf.
20. David Jao and Luca De Feo. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. In Bo-Yin Yang, editor, *Post-Quantum Cryptography*, pages 19–34, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.
21. P. Kutas, Simon-Philipp Merz, C. Petit, and Charlotte Weitkämper. One-way functions and malleability oracles: Hidden shift attacks on isogeny-based protocols. *IACR Cryptol. ePrint Arch.*, 2021:282, 2021.
22. Christopher Leonardi. A note on the ending elliptic curve in sidh. Cryptology ePrint Archive, Report 2020/262, 2020. https://eprint.iacr.org/2020/262.

23. Tomoki Moriya, Hiroshi Onuki, and Tsuyoshi Takagi. SiGamal: A Supersingular Isogeny-Based PKE and Its Application to a PRF. In Shiho Moriai and Huaxiong Wang, editors, *Advances in Cryptology – ASIACRYPT 2020*, pages 551–580, Cham, 2020. Springer International Publishing.
24. National Institute of Standards and Technology: Post quantum Cryptography Standardization, December 2016. https://csrc.nist.gov/Projects/Post-Quantum-Cryptography/Post-Quantum-Cryptography-Standardization.
25. Chris Peikert. He Gives C-Sieves on the CSIDH. *Advances in Cryptology – EUROCRYPT 2020*, 12106:463 – 492, 2019.
26. Christophe Petit. Faster algorithms for isogeny problems using torsion point images. In Tsuyoshi Takagi and Thomas Peyrin, editors, *Advances in Cryptology – ASIACRYPT 2017*, pages 330–353. Springer International Publishing, 2017.
27. A. Rostovtsev and A. Stolbunov. Public-key cryptosystem based on isogenies. *IACR Cryptol. ePrint Arch.*, 2006:145, 2006.
28. Joseph H Silverman. *The arithmetic of elliptic curves*, volume 106. Springer Science & Business Media, 2009.
29. The Sage Developers. *SageMath, the Sage Mathematics Software System (Version 9.0)*, 2020. `https://www.sagemath.org`.
30. David Urbanik and David Jao. New techniques for SIDH-based NIKE. *Journal of Mathematical Cryptology*, 14(1):120–128, 2020.

A HealS PKE

The HealS Public Key Encryption scheme is detailed in Figure 4.

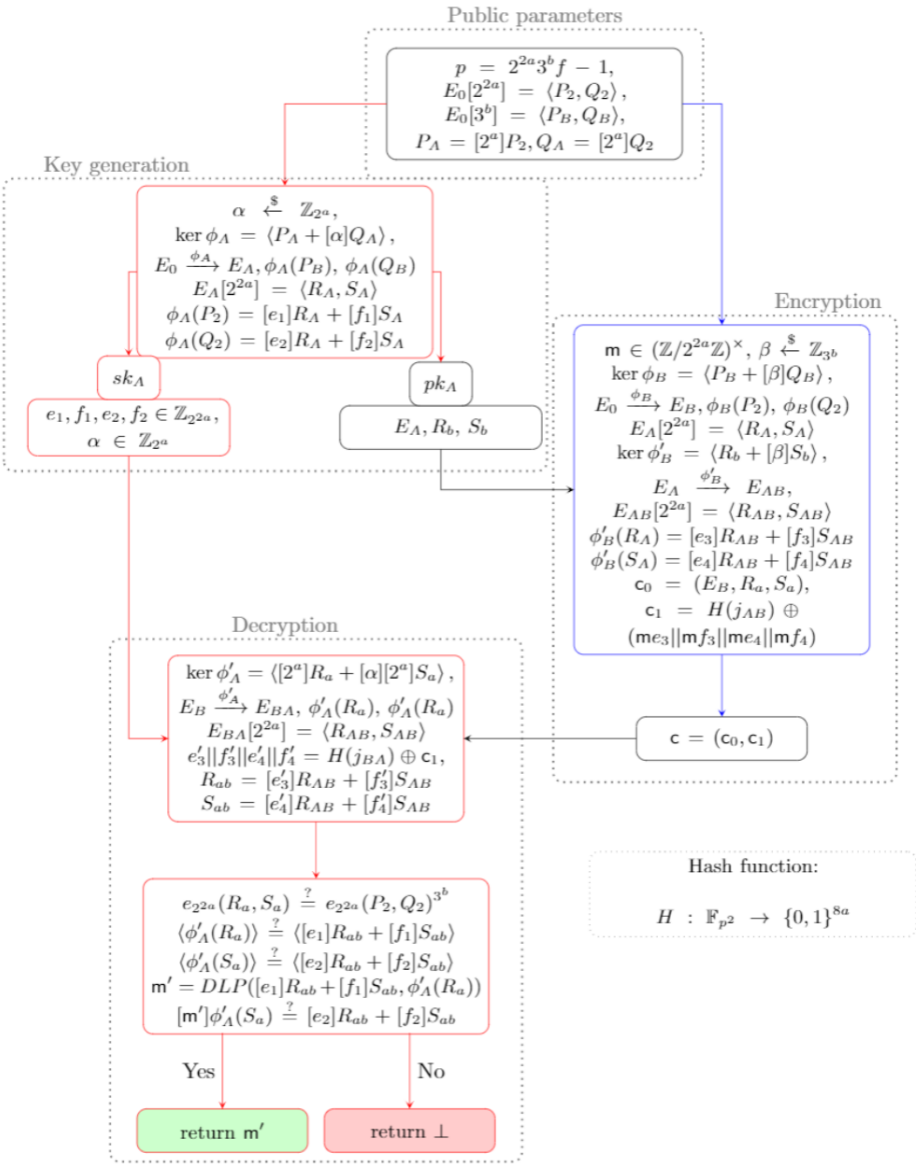

Fig. 4: HealS PKE.