Attack on SHealS and HealS: the Second Wave of GPST

Steven D. Galbraith[1] and Yi-Fu Lai[1]

[1]University of Auckland, New Zealand

s.galbraith@auckland.ac.nz ylai276@aucklanduni.ac.nz

7th April 2022

Abstract

We cryptanalyse the SHealS and HealS cryptosystems of Fouotsa and Petit from Asiacrypt 2021.

1 Introduction

An efficient and secure static-static public key encryption has been a long-standing desired primitive in the isogeny community. It has been known that CSIDH [CLM+18] with a slightly different algebraic structure from SIDH [JD11] can give a straightforward and efficient option and can be used to realize several practical and competitive cryptographic primitives from isogenies [MOT20, LGd21, LD21, BDK+21]. However, the algebraic structure also make them falls prey to a subexponential attack – the Kuperberg algorithm [Kup05]. Hence, an efficient protocol with a robust underlying assumption is always a more attractive option.

The main bottleneck for SIDH-family schemes to achieve the notion boils down to the adaptive GPST attack [GPST16] which enable malicious Bob to bit-wisely extract Alice's secret key from each handshake, and vice versa. The known countermeasures against the attack are to embedd zero-knowledge proof [UJ20] or to utilize the k-SIDH method [AJL17]. However, these countermeasures also incurred multiple parallel isogeny computations so that the products cannot be practical. To resolve this, Fouotsa and Petit [FP21] (Asiacrypt'21) presented a variant of SIDH with a devised key validation mechanism. The scheme takes less number of isogeny computation than SIKE [ACC+17] with parameter size doubled in length which still is far more efficient than other known abovementioned solutions. They conjecture and claim the scheme gives a static-static encryption solution from isogenies immune to any adaptive attacks.

In this work we refute the claim by presenting an adaptive attack against their protocol. Our attack builds on their key validation mechanism which relies on a secret to execute. The attack can be viewed as a simple tweak of the GPST attack and, surprisingly, it takes the same number of oracle queries as the GPST attack against SIDH to adaptively recover a secret key. In other words, the additional mechanism not only slows down the protocol but also gives no advantage to the scheme for preventing the adaptive attack.

1.1 Technique overview

The cornerstone of our attack is the flaw orginating in the proof their main theorem for the key validation mechanism (Theorem 2 in [FP21]). Their key validation mechanism consists of following three equations:

$$\begin{aligned} e_{4^a}(R_a, S_a) &= e_{4^a}(P_2, Q_2)^{3^b}, \\ \psi'_A(R_a) &= [e_1]R_{ab} + [f_1]S_{ab} \in E_{AB}, \\ \psi'_A(S_a) &= [e_2]R_{ab} + [f_2]S_{ab} \in E_{AB}, \end{aligned}$$

where ψ'_A is an isogeny from E_B with kernel $\langle [2^a]R_a + [\alpha 2^a]S_a \rangle \in E_B$, $(R_a, S_a, R_{ab}, S_{ab}, E_B, E_{AB})$ is given by Bob, and $(\alpha, e_1, f_1, e_2, f_2)$ is Alice's secret key.

These relations will be satisfied when Bob produces the input honestly. In their analysis, to make another valid input except for taking negations on the curve points is equivalent to solve four linear equations with four unknown variables (e_1, f_1, e_2, f_2) over the ring $\mathbb{Z}/4^a\mathbb{Z}$. Furthermore, the input of Bob also has restriction that $e_{4^a}(R_a, S_a) = e_{4^a}(P_2, Q_2)^{3^b}$ and ψ'_A might vary with the choice of R_a and S_a. Therefore, they conclude that Bob, without knowing the secret, is not able to produce another valid input except for taking negations on the original input. In this way, since Bob, restricted by the mechanism, behaves honestly, the cryptosystem will be secure based on the hardness assumption.

However, for an adaptive attack, what malicious Bob want to exploit is that Alice's behaviour is dependent on the secret. The proof neglects the spirit of the adaptive attack where malicious Bob can learn the information adaptively. For example, write $\mathbf{M} = \begin{pmatrix} e_1 & f_1 \\ e_2 & f_2 \end{pmatrix} \in M_{2\times 2}(\mathbb{F}_{4^a}), \mathbf{u} = (R_a\ S_a)^T$ and $\mathbf{v} = (R_{ab}\ S_{ab})^T$. We may therefore abuse the notation by writing $\psi'_A\mathbf{u} = \mathbf{M}\mathbf{v}$. As we will show in Sec. 3, we find a pair of special matrices $\mathbf{P}_1 = \begin{pmatrix} 1 & 0 \\ 2^{2a-1} & 1 \end{pmatrix}$ and $\mathbf{P}_2 = \mathbf{I}_2$ such that the commutativity of $\mathbf{P}_1\mathbf{M} = \mathbf{M}\mathbf{P}_2$ holds if and only if $e_1 = f_1 = 0 \mod 2$. Hence, on input $(R'_a, S'_a, R'_{ab}, S'_{ab}, E_B, E_{AB})$ where $(R'_a\ S'_a)^T = \mathbf{P}_1\mathbf{u}$ and $(R'_{ab}\ S'_{ab})^T = \mathbf{P}_2\mathbf{v}$ the key validation mechanism will pass if and only if $\psi'_A\mathbf{P}_1\mathbf{u} = \mathbf{M}\mathbf{P}_2\mathbf{v}$ if and only if $e_1 = f_1 = 0 \mod 2$. Note that because $\det(\mathbf{P}_1) = 1$ and $(2^a\ \alpha 2^a)\mathbf{P}_1 = (c\ c)$ for some $c \in \mathbb{Z}_{2^a}$, the Weil pairing check will also pass and the isogeny used by the mechanism is still ψ'_A. In this way, Bob learns 1-bit information of e_1 and f_1. Moreover, as we will show in Sec. 3, this is enough to recover the least significant bit of α.

On top of that, Bob can utilize the GPST attack in a "reciprocal" sense to extract more information further. If the significant bit $\alpha_0 = 1$, the secret α is invertible over the ring $\mathbb{Z}/2^a\mathbb{Z}$. By further replacing R_a with $R'_a = R_a + [2^{2a-2}]R_a - [2^{2a-2}\alpha_0]S_a$. The equality of the second equation depends on the second least significant bit of α. However, $e_{4^a}(R'_a, S_a)$ will never satisfy the first equation. To overcome this, Bob will replace S_a with $[\alpha_0^{-1}2^{2a-2}]R_a + [1 - 2^{2a-2}]S_a$, which is used to extract the second least significant bit of α^{-1}, because the equality of the third equation depends on the second least significant bit of α^{-1}. Remark that, the isogeny used in the key validation mechanism is still ψ'_A since the kernel is $\langle [2^a]R_a + [\alpha 2^a]S_a \rangle$. In Sec. 4, we present the attack in details including the case that α is even.

Structure of this Paper. We begin in Sec. 3 with some preliminary backgrounds on elliptic curves, isogenies, a breif outline of their HealSIDH scheme, together with a few immediate propositions of the scheme derived from the parameter. We then introduce the method of using commutativity of matrices to extract the least significant bit of Alice's secret in Sec. 3. Based on the least significant bit information, a tweak of the GPST attack to recursively and adaptively recover Alice's secret is presented in the proceeding sections: Sec. 4. A breif summary is made in Sec. 5. We also provide in App. A a generic attack against their framework with the key validation mechanism where the primes 2 and 3 are replaced by other small primes.

2 Preliminaries

Notations. We begin by introducing some notations that will be used throughout the paper. Let $\mathbf{O}$ represent the point at infinity, $\mathbb{N}$ be the set of natural numbers, and $\mathbb{Z}$ be the set of integers. For $n \in \mathbb{N}$, let $\mathbb{Z}_n$ defined to be $\mathbb{Z}/n\mathbb{Z}$ and $\mathbb{F}_n$ be the finite field of order n. For convenience, when we write $u \in \mathbb{Z}_n$, we consider u is a representative taken from $\{0, \cdots, n-1\} \subset \mathbb{Z}$. Similarly, when we write $u \mod n$, we consider the value is taken from $\{0, \cdots, n-1\} \subset \mathbb{Z}$.

2.1 Elliptic curves and isogenies

An elliptic curve is a rational nonsingular curve of genus one with a distinguished point at infinity denoted by $\mathbf{O}$. An elliptic curve with $\mathbf{O}$ forms an additive commutative group. If E is an elliptic curve defined over $\mathbb{F}_p$, then $E(\mathbb{F}_p)$, collecting $\mathbb{F}_p$-rational points of E, is a finite subgroup of E. Moreover, E is said to be supersingular if the cardinality of $E(\mathbb{F}_p)$ is $p+1$. For $n \in \mathbb{N}$ coprime with p, the n-torsion subgroup $E[n]$, collecting points of order dividing n, is isomorphic to $\mathbb{Z}_n \oplus \mathbb{Z}_n$.

An isogeny is a morphism between elliptic curves preserving the point at infinity. The kernel of an isogeny is always finite and defines the isogeny up to a power of the Frobenius map. We restrict our attention to separable isogenies (which induce separable extensions of function fields over $\mathbb{F}_p$) between supersingular elliptic curves defined over $\mathbb{F}_p$. Given a finite subgroup S of E, there exists a unique separable isogeny with kernel S from E to the codomain denoted by E/S via Vélu's formulas. We refer [Sil09] to get more exposed to the elliptic curve theory.

2.2 Brief outline of HealSIDH

Both SHealS and HealS of [FP21] are PKE schemes building on the key exchange scheme of HealSIDH with a devised key validation mechanism. We breifly introduce a outline of HealSIDH with the key validation mechanism. The public parameter $\mathsf{pp} = (E_0, P_2, Q_2, P_3, Q_3, p, a, b)$ contains a supersingular curve E_0 with an unknown endomorphism ring and $(p, a, b) \in \mathbb{N}^3$ where a prime $p = 2^{2a}3^{2b}f - 1$ such that $2^a \approx 3^b$. The set $\{P_2, Q_2\}$, $\{P_3, Q_3\}$ are bases for $E_0[4^a]$ and $E_0[9^b]$ respectively and $P_A = [2^a]P_2, Q_A = [2^a]Q_2, P_B = [3^b]P_3$, and $Q_B = [3^b]Q_3$. Alice and Bob sample α and β uniformly at random from $\mathbb{Z}_{2^a}$ and $\mathbb{Z}_{3^b}$ respectively. Also, Alice and Bob compute $\phi_A : E_0 \to E_A = E_0/\langle P_A + [\alpha]Q_A\rangle$ and $\phi_B : E_0 \to E_A = E_0/\langle P_B + [\beta]Q_B\rangle$, respectively. Alice and Bob compute $(\phi_A(P_2), \phi_A(Q_2), \phi_A(P_B), \phi_A(Q_B))$ and $(\phi_B(P_3), \phi_B(Q_3), \phi_B(P_A), \phi_B(Q_A))$ respectively. Alice's and Bob's public keys are $(E_A, \phi_A(P_3), \phi_A(Q_3))$ and $(E_B, \phi_B(P_2), \phi_B(Q_2))$ respectively. Alice computes the canonical basis $\{R_A, S_A\}$ for $E_A[4^a]$ and represents $\phi_A(P_2) = [e_1]R_A + [f_1]S_A$ and $\phi_A(Q_2) = [e_2]R_A + [f_2]S_A$. Bob computes the canonical basis $\{R_B, S_B\}$ for $E_B[9^a]$ and represents $\phi_B(P_3) = [g_1]R_B + [h_1]S_B$ and $\phi_B(Q_3) = [g_2]R_B + [h_2]S_B$. Alice's and Bob's secret keys are $\mathsf{sk}_A = (\alpha, e_1, f_1, e_2, f_2)$ and $\mathsf{sk}_B = (\beta, g_1, h_1, g_2, h_2)$ respectively.

To establish a shared secret with Alice, he collects Alice public key, denoted by (E_A, R_b, S_b), and 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 $\phi'_B(S_b))$. He sends $(R_{ab} = \phi'_B(R_A), S_{ab} = \phi'_B(S_A))$ to Alice.

Upon receiving (R_{ab}, S_{ab}) from Bob, Alice collects Bob's public key (E_B, R_a, S_a). She computes $\psi'_A : E_B \to E_{BA} = E_B/\langle [3^b]R_a + [\beta 3^b]S_a\rangle$ together with $(\psi'_A(R_B), \psi'_A(S_B), \psi'_A(R_a), \psi'_A(S_a))$. If $e_{4^a}(R_a, S_a) \neq e_{4^a}(P_2, Q_2)^{3^b}$, $\psi'_A(R_a) \neq [e_1]R_{ab} + [f_1]S_{ab}$, or $\psi'_A(S_a) \neq [e_2]R_{ab} + [f_2]S_{ab}$, then Alice aborts (the session). Otherwise, she sends $(R_{ba} = \phi'_A(R_B), S_{ba} = \phi'_A(S_B))$ to Bob and keeps the j-invariant j_{BA} of E_{BA} as the shared secret.

Similarly, upon receiving (R_{ba}, S_{ba}), Bob aborts if $e_{9^a}(R_b, S_b) \neq e_{9^a}(P_3, Q_3)^{2^a}$, $\phi'_B(R_b) \neq [g_1]R_{ba} + [h_1]S_{ba}$, or $\phi'_B(S_b) \neq [g_2]R_{ba} + [h_2]S_{ba}$, If not he takes the j-invariant of E_{AB} as the shared secret.

Remark 2.1. *Remark that for the issue of eavesdropping security, Bob will give the coordinates of R_{ab}, S_{ab} with respect to the canonical basis of $E_B[4^a]$. Otherwise, the secretly shared curve E_{AB} can be recontructed*

by an eavesdropper. Nonetheless, it does not matter in our attack model because (honest) Alice will get the points eventually and (malicious) Bob does not require R_{ba}, S_{ba} to attack. Hence, for the convenience, we may assume Bob sends the entire point R_{ab}, S_{ab} to Alice.

We can have following two immediate results.

Proposition 2.2. *If Bob honestly generate $R_a = \phi_B(P_2)$, $S_a = \phi_B(Q_2)$, $R_{ab} = \phi'_B(R_A)$ and $S_{ab} = \phi'_B(S_A)$, then $\{R_{ab}, S_{ab}\}$ is a basis of $E_{AB}[4^a]$ and $\{R_a, S_a\}$ is a basis of $E_B[4^a]$.*

Proof. Since $[4^a]R_a = \phi_B([4^a]P_2) = \mathbf{O}$ and $[4^a]S_a = \phi'_B([4^a]Q_2) = \mathbf{O}$, both R_a and S_a are in $E_B[4^a]$. Due to $e_{4^a}(R_a, S_a) = e_{4^a}(P_2, Q_2)^{3^b}$, we know $e_{4^a}(R_a, S_a)$ is a primitive 4^a-th root of unity. Similarly, since $[4^a]R_{ab} = \phi'_B([4^a]R_A) = \mathbf{O}$ and $[4^a]S_{ab} = \phi_B([4^a]S_A) = \mathbf{O}$, both R_{ab} and S_{ab} are in $E_{AB}[4^a]$. Due to $e_{4^a}(R_{ab}, S_{ab}) = e_{4^a}(R_A, S_A)^{3^b}$, we know $e_{4^a}(R_{ab}, S_{ab})$ is a primitive 4^a-th root of unity. Therefore, the result follows. □

Lemma 2.3. *Let e_1, e_2, f_1, f_2 defined as above and $\alpha \in \mathbb{Z}_{2^a}$ be the secret key of Alice such that $\ker(\phi_A) = \langle [2^a]P_2 + [\alpha 2^a]Q_2 \rangle$. If Alice follows the protocol specification, then $e_1 + \alpha e_2 = f_1 + \alpha f_2 = 0 \mod 2^a$.*

Proof. Given $\phi_A(P_2) = [e_1]R_A + [f_1]S_A$ and $\phi_A(Q_2) = [e_2]R_A + [f_2]S_A$, we have $\mathbf{O} = \phi_A([2^a]P_2 + [\alpha 2^a]Q_2) = [2^a e_1 + \alpha 2^a e_2]R_A + [2^a f_1 + \alpha 2^a f_2]S_A = [e_1 + \alpha e_2]([2^a]R_a) + [f_1 + \alpha f_2]([2^a]S_A)$.

Recall that $\{[2^a]R_A, [2^a]S_A\}$ is a basis for $E_A[2^a]$ due to $\{R_A, S_A\}$ being a basis for $E_A[4^a]$. Therefore, $e_1 + \alpha e_2 = f_1 + \alpha f_2 = 0 \mod 2^a$.

□

Modeling. Throughout this paper, we consider adaptive attack against HealSIDH. Bob, as an adversary, is given access to an oracle $\mathcal{O}_{\mathsf{sk}_A} \to 0/1$ taking as input $(R_a, S_a, R_{ab}, S_{ab}, E_B, E_{AB})$ with the relation specified as above. For simplicity, we denote the oracle by $\mathcal{O}$ and omit the inputs of curves E_B, E_{AB} without causing confusion. The oracle returns 1 if and only if the following three equations hold:

$$e_{4^a}(R_a, S_a) = e_{4^a}(P_2, Q_2)^{3^b}, \tag{1}$$

$$\psi'_A(R_a) = [e_1]R_{ab} + [f_1]S_{ab}, \tag{2}$$

$$\psi'_A(S_a) = [e_2]R_{ab} + [f_2]S_{ab}, \tag{3}$$

where ψ'_A is an isogeny from E_B with kernel $\langle [2^a]R_a + [\alpha 2^a]S_a \rangle \in E_B$.

When Bob follows the protocol specification, the three equations hold naturally. The goal of malicious Bob in our attack is to recover the secret α in sk_A by adaptively manipulating the oracle.

3 Parity Recovering

In this section, we consider the least significant bits of e_1, e_2, f_1, f_2 and α. We can recover the least significant bit of α with one oracle query by relying the relations given by Lem. 2.3.

The attack presented in this section and the next section relies on following facts:

- $\{P_2, Q_2\}$, is a basis for $E_0[4^a]$.
- $\{R_{ab}, S_{ab}\}$ is a basis of $E_{AB}[4^a]$ (Prop. 2.2).
- $\{R_a, S_a\}$ is a basis of $E_B[4^a]$ (Prop. 2.2).
- $e_1 + \alpha e_2 = f_1 + \alpha f_2 = 0 \mod 2^a$ (Lem. 2.3).

The high-level idea in this section is simple. Assume Alice and Bob follows the protocol specification. Write $\mathbf{M} = \begin{pmatrix} e_1 & f_1 \\ e_2 & f_2 \end{pmatrix} \in M_{2\times 2}(\mathbb{F}_{4^a}), \mathbf{u} = (R_a\ S_a)^T$ and $\mathbf{v} = (R_{ab}\ S_{ab})^T$. Recall that $\psi'_A(R_a) = [e_1]R_{ab} + [f_1]S_{ab}$, $\psi'_A(S_a) = [e_2]R_{ab} + [f_2]S_{ab}$ where R_a, S_a, R_{ab}, S_{ab} are honestly generated by Bob. We may abuse the notation by writing $\psi'_A\mathbf{u} = \mathbf{M}\mathbf{v}$ based on Eqs. (2) and (3). The idea is to find a pair of particular square matrices $\mathbf{P}_1, \mathbf{P}_2 \in M_{2\times 2}(\mathbb{F}_{4^a})$ where $\mathbf{P}_1$ is of determinant 1 such that the commutativity of $\mathbf{P}_1\mathbf{M} = \mathbf{M}\mathbf{P}_2$ is conditioned on the information (parity for instance) to be extracted from $\mathbf{M}$. Let $(R'_a\ S'_a)^T = \mathbf{P}_1\mathbf{u}$ and $(R'_{ab}\ S'_{ab})^T = \mathbf{P}_2\mathbf{v}$. On input $(R'_a, S'_a, R'_{ab}, S'_{ab})$ the oracle returns 1 if $\mathbf{M}$ satisfies the condition of the commutativity $\mathbf{P}_1\mathbf{M} = \mathbf{M}\mathbf{P}_2$, because $\psi'_A\mathbf{P}_1\mathbf{u} = \mathbf{P}_1\mathbf{M}\mathbf{v} = \mathbf{M}\mathbf{P}_2\mathbf{v}$ holds. Remark that the determinant 1 of $\mathbf{P}_1$ ensures the new pair $(R'_a\ S'_a)$ will satisfy the Weil pairing verification Eq. (1). Futhermore, we require $(2^a\ \alpha 2^a)\mathbf{P}_1 = (c\ c)$ for some $c \in \mathbb{Z}_{2^a}$ so that the isogeny used by the oracle is still the one with the kernel $\langle [2^a]R_a + [\alpha 2^a]S_a \rangle$.

Though there are 2^4 combinations of the least significant bits of e_1, e_2, f_1, f_2. The following lemma shows that when Alice generates them honestly, there are only six patterns.

Lemma 3.1. *If Alice produces $\phi_A(P_2)$ and $\phi_A(Q_2)$ honestly, then there are only 6 possible patterns of parities of e_1, e_2, f_1, f_2:*

1. $f_2 = 1 \mod 2$ *and* $e_2 = e_1 = f_1 = 0 \mod 2$
2. $e_2 = 1 \mod 2$ *and* $e_1 = f_1 = f_2 = 0 \mod 2$,
3. $f_2 = e_2 = 1 \mod 2$ *and* $e_1 = f_1 = 0 \mod 2$,
4. $f_1 = f_2 = 1 \mod 2$ *and* $e_2 = e_1 = 0 \mod 2$,
5. $e_2 = e_1 = 1 \mod 2$ *and* $f_1 = f_2 = 0 \mod 2$,
6. $e_1 = f_1 = e_2 = f_2 = 1 \mod 2$.

Proof. Recall $e_{4^a}(\phi_A(P_2), \phi_A(Q_2)) = e_{4^a}(P_2, Q_2)^{2^a} = e_{4^a}(R_A, S_A)^{e_1 f_2 - e_2 f_1}$. Since both $\{P_2, Q_2\}$ and $\{R_A, S_A\}$ are bases for $E_0[4^a], E_A[4^a]$ respectively, both $e_{4^a}(P_2, Q_2)$ and $e_{4^a}(R_A, S_A)$ are primitive 4^a-th roots of unity. Given $e_{4^a}(R_A, S_A)^{2^a(e_1 f_2 - e_2 f_1)} = 1$, we have $e_1 f_2 - e_2 f_1 = 0 \mod 2^a$.

Furthermore, e_2, f_2 cannot be both even. Recall $\phi(Q_2) = e_2 R_A + f_2 S_A$. Suppose for the purpose of contradiction that both e_2 and f_2 are even. Then, $[4^{a-1}]\phi_A(Q_2) = \mathbf{O}$, which implies $\ker(\phi_A) = \langle P_2 + [\alpha]Q_2 \rangle$ contains $[4^{a-1}]Q_2$. That is, $[k]P_2 + [k\alpha]Q_2 = [4^{a-1}]Q_2$ for some $k \in \mathbb{Z}_{2^a}$, so $k = 0$. This contradicts the fact that $\{P_2, Q_2\}$ is a basis for $E_0[4^a]$. The result follows. □

We order the six cases according to the lemma above. The following lemmata indicate that we can divide the overall cases into two partitions: {*Case* 1, *Case* 2, *Case* 3} and {*Case* 4, *Case* 5, *Case* 6} with 1 oracle query.

Lemma 3.2. *Assume Bob honestly generates $R_a, S_a, R_{ab}, S_{ab}, E_B, E_{AB}$. On input $(R_a, [2^{2a-1}]R_a + S_a, R_{ab}, S_{ab})$, the oracle returns* 1 *only for Cases 1 to 3.*

Proof. Firstly, the isogeny ψ'_A computed by the oracle is the same one used by Alice in the honest execution. This is because both kernels are the same:

$$\langle [2^a]R_a + [\alpha 2^a]S_a \rangle = \langle [2^a]R_a + [\alpha 2^a]([2^{2a-1}]R_a + S_a) \rangle.$$

Therefore, since R_a, S_a, R_{ab}, S_{ab} are honestly generated, we may assume $e_{4^a}(R_a, S_a) = e_{4^a}(P_2, Q_2)^{3^b}$, $\psi'_A(R_a) = [e_1]R_{ab} + [f_1]S_{ab}$, and $\psi'_A(S_a) = [e_2]R_{ab} + [f_2]S_{ab}$.

For Eq. (1), since $e_{4^a}(R_a, S_a) = e_{4^a}(P_2, Q_2)^{3^b}$, we have

$$e_{4^a}(R_a + [2^{a-1}]S_a, S_a) = e_{4^a}(R_a, S_a) = e_{4^a}(P_2, Q_2)^{3^b}.$$

Let $e'_1, e'_2, f'_1, f'_2 \in \{0, 2^{2a-1}\}$ denote the parity bits of e_1, e_2, f_1, f_2 left-shifted by $2a-1$ bits, resp. Given $\psi'_A(R_a) = [e_1]R_{ab} + [f_1]S_{ab}$, $\psi'_A(S_a) = [e_2]R_{ab} + [f_2]S_{ab}$ and $R_{ab}, S_{ab} \in E_{AB}[2^a]$, we have

$$\psi'_A(R_a) - [e_1]R_{ab} - [f_1]S_{ab} = \mathbf{O},$$
$$\psi'_A([2^{2a-1}]R_a + S_a) - [e_2]R_{ab} - [f_2]S_{ab} = [e'_1]R_{ab} + [f'_1]S_{ab}.$$

Recall that $\{R_{ab}, S_{ab}\}$ is a basis. Therefore, the oracle returns 1 if and only if $[e'_1]R_{ab} + [f'_1]S_{ab} = \mathbf{O}$ or, equivalently, $e'_1 = f'_1 = 0$. The result follows. □

Lemma 3.3. *Assume Bob honestly generates $R_a, S_a, R_{ab}, S_{ab}, E_B, E_{AB}$. On input $([1+2^{2a-1}]R_a - [2^{2a-1}]S_a, [2^{2a-1}]R_a + [1-2^{2a-1}]S_a, R_{ab}, S_{ab})$, the oracle returns 1 only for Cases 4 to 6.*

Proof. Firstly, the isogeny ψ'_A computed by the oracle is the same one used by Alice in the honest execution. This is because both kernels are the same:

$$\langle [2^a]R_a + [\alpha 2^a]S_a \rangle = \langle [2^a]([1+2^{2a-1}]R_a - [2^{2a-1}]S_a) + [\alpha 2^a]([2^{2a-1}]R_a + [1-2^{2a-1}]S_a) \rangle.$$

Therefore, since R_a, S_a, R_{ab}, S_{ab} are honestly generated, we may assume $e_{4^a}(R_a, S_a) = e_{4^a}(P_2, Q_2)^{3^b}$, $\psi'_A(R_a) = [e_1]R_{ab} + [f_1]S_{ab}$, and $\psi'_A(S_a) = [e_2]R_{ab} + [f_2]S_{ab}$.

For Eq. (1), since $e_{4^a}(R_a, S_a) = e_{4^a}(P_2, Q_2)^{3^b}$, we have

$$\begin{aligned} & e_{4^a}([1+2^{a-1}]R_a - [2^{a-1}]S_a, [2^{a-1}]R_a + [1-2^{a-1}]S_a) \\ = \; & e_{4^a}(R_a, S_a)^{1-2^{2a-2}+2^{2a-2}} \\ = \; & e_{4^a}(P_2, Q_2)^{3^b}. \end{aligned}$$

Let $e'_1, e'_2, f'_1, f'_2 \in \{0, 2^{2a-1}\}$ denote the parity bits of e_1, e_2, f_1, f_2 left-shifted by $2a-1$ bits, resp. Given $\psi'_A(R_a) = [e_1]R_{ab} + [f_1]S_{ab}$, $\psi'_A(S_a) = [e_2]R_{ab} + [f_2]S_{ab}$ and $R_{ab}, S_{ab} \in E_{AB}[2^a]$, we have

$$\psi'_A([1+2^{2a-1}]R_a - [2^{2a-1}]S_a) - [e_1]R_{ab} - [f_1]S_{ab} = [e'_1]R_{ab} + [f'_1]S_{ab} + [e'_2]R_{ab} + [f'_2]S_{ab},$$
$$\psi'_A([2^{2a-1}]R_a + [1-2^{2a-1}]S_a) - [e_2]R_{ab} - [f_2]S_{ab} = [e'_1]R_{ab} + [f'_1]S_{ab} + [e'_2]R_{ab} + [f'_2]S_{ab}.$$

Recall that $\{R_{ab}, S_{ab}\}$ is a basis. Therefore, the oracle returns 1 if and only if $e'_1 = e'_2$ and $f'_1 = f'_2$. The result follows. □

One can observe that it is one of the case in $\{Case\ 1,\ Case\ 2,\ Case\ 3\}$ if and only if the least significant bit of α is 0 by Lem. 3.1. In fact, by choosing particular matrices $\mathbf{P}_1$ and $\mathbf{P}_2$, one can precisely recover all e'_1, e'_2, f'_1 and f'_2. However, by Lem. 3.1, we do not bother to find them all since the information given in Lem. 3.2 already is sufficient to recover the least significant bit of α. In the next section, we will start with the least significant bit of α to recover each higher bit with one oracle query for each.

4 Recover the Secret

In this section, we present a variant of the GPST attack to recover the secret α based on the knowledge extracted from the previous section. The high-level idea is to use the GPST attack in a "reciprocal" manner. Recall that Bob has two following equations when he generates the points $(R_a, S_a, R_{ab}, S_{ab})$ honestly:

$$\begin{aligned} \psi'_A(R_a) &= [e_1]R_{ab} + [f_1]S_{ab}, \\ \psi'_A(S_a) &= [e_2]R_{ab} + [f_2]S_{ab}, \end{aligned}$$

where $\ker(\psi'_A) = \langle [2^a]R_a + [2^a\alpha]S_a \rangle$.

To extract the second least significant bit of α, denoted by α_1, based on the least bit α_0, we consider $\psi'_A(R_a + [2^{2a-2}]R_a - [2^{2a-2}\alpha_0]S_a) = [e_1]R_{ab} + [f_1]S_{ab} + ([2^{2a-2}e_1 - 2^{2a-2}\alpha_0 e_2]R_{ab} + [2^{2a-2}f_1 - 2^{2a-2}\alpha_0 f_2]S_{ab})$

where the purpose of $[2^{2a-2}\alpha_0]S_a$ is to eliminate the lower bit. Note that $([2^{2a-2}e_1 - 2^{2a-2}\alpha_0 e_2]R_{ab} + [2^{2a-2}f_1 - 2^{2a-2}\alpha_0 f_2]S_{ab}) = ([\alpha_1 2^{2a-1}][e_2]R_{ab} + [\alpha_1 2^{2a-1}][f_2]S_{ab})$ because $e_1 + \alpha e_2 = f_1 + \alpha f_2 = 0 \mod 2^a$ and $\{R_a, S_a\}$ is a basis for $E_B[4^a]$ (Lem. 2.3 and Prop. 2.2). By Lem. 3.1, since e_2 and f_2 cannot be both even, at least one of $[2^{2a-1}e_2]R_{ab}$ and $[2^{2a-1}f_2]S_{ab}$ is of order 2. It follows that the equation

$$\psi'_A(R_a + [2^{2a-2}]R_a - [2^{2a-2}\alpha_0]S_a) = [e_1]R_{ab} + [f_1]S_{ab}$$

holds if and only if $\alpha_1 = 0$.

Unfortunately, giving $(R_a + [2^{2a-2}]R_a - [2^{2a-2}\alpha_0]S_a, S_a, R_{ab}, S_{ab})$ to Alice cannot work, because $e_{4^a}(R_a + [2^{2a-2}]R_a - [2^{2a-2}\alpha_0]S_a, S_a)$ never equals $e_{4^a}(P_2, Q_2)^{3^b}$. In other words, if Bob does so, he will always get $\perp$ from Alice. To resolve this, we use the idea of "reciprocal". Assume α is invertible modulo 2^a. Bob will craft a point replacing S_a for *recovering* $\alpha^{-1} \mod 2^a$ *at the same time*. Concretely, Bob computes $\hat{\alpha} = \alpha_0^{-1} \mod 4$ as if α_1 would be 0 (might not be true). For the same reasoning as above, the equation

$$\psi'_A(\hat{\alpha}[2^{2a-2}]R_a + [1 - 2^{2a-2}]S_a) = [e_2]R_{ab} + [f_2]S_{ab}$$

holds if and only if $\alpha^{-1} = \hat{\alpha} \mod 4$ if and only if $\alpha_1 = 0$.

Moreover, $e_{4^a}(R_a + [2^{2a-2}]R_a - [2^{2a-2}\alpha_0]S_a, \hat{\alpha}[2^{2a-2}]R_a + [1 - 2^{2a-2}]S_a) = e_{4^a}(R_a, S_a)$. Therefore, by sending $(R_a + [2^{2a-2}]R_a - [2^{2a-2}\alpha_0]S_a, \hat{\alpha}[2^{2a-2}]R_a + [1 - 2^{2a-2}]S_a, R_{ab}, S_{ab})$ to Alice, Bob can know whether $\alpha_1 = 0$. However, α is not necessarily to be even. We have to use unbalanced powers of 2 on each query and introduce the concept of *quasi-inverse* elements.

Remark 4.1. *On input* $(R_a + [2^{2a-2}]R_a - [2^{2a-2}\alpha_0]S_a, \hat{\alpha}[2^{2a-2}]R_a + [1 - 2^{2a-2}]S_a, R_{ab}, S_{ab})$*, honest Alice will use the same isogeny* ψ'_A *because* $\langle [2^a](R_a + [2^{2a-2}]R_a - [2^{2a-2}\alpha_0]S_a) + [\alpha 2^a](\hat{\alpha}[2^{2a-2}]R_a + [1 - 2^{2a-2}]S_a)\rangle = \langle [2^a]R_a + [\alpha 2^a]S_a\rangle$*. As a result, the same kernel will derive the same isogeny* ψ'_A*.*

4.1 Quasi-Inverse Element

Definition 4.2. *Let* p *be a prime and* $a \in \mathbb{N}$*. For an element* $u \in \mathbb{Z}$*, a* p^a-quasi-inverse element *of* u *is a non-zero element* $v \in \mathbb{Z}_{p^a}$ *such that* $uv = p' \mod p^a$ *where* p' *is the maximal power of* p *dividing* u*.*

When $a = 1$, every element obviously has a *p-quasi-inverse element* by taking either its inverse over $\mathbb{Z}_p$ or 1. Unlike the inverse over a ring, a quasi-inverse is not necessarily unique. For instance, 9 and 17 are 2^5-quasi-inverse elements of 4 over $\mathbb{Z}_{32}$. A non-zero element being not a unit of $\mathbb{Z}_{p^a}$ can still have a p^a-quasi-inverse element. However, a non-zero element v in $\mathbb{Z}_{p^a}$ being a p^a-quasi-inverse element for a non-zero integer in $\mathbb{Z}_{p^a}$ implies v is a unit of $\mathbb{Z}_{p^a}$.

Proposition 4.3. *Let* p *be a prime and* $a \in \mathbb{N}$*. For* $u \in \mathbb{Z}$*, a non-zero element over* $\mathbb{Z}_{p^a}$*, any* p^a*-quasi-inverse element of* u *is a unit of* $\mathbb{Z}_{p^a}$*.*

Proof. Write $u = u'p^j$ where $u', j \in \mathbb{Z}$ and u' is not divisible by p and $j < a$. Say there exists $v \in \mathbb{Z}_{p^a}$ such that $uv = p^j \mod p^a$. Since u is a non-zero element over $\mathbb{Z}_{p^a}$, we know $a > j$ so that $(u/p^j)v = 1 \mod p^{j-a}$. It follows that v is not devided by p, so v is a unit of $\mathbb{Z}_{p^a}$. □

In fact, for any $u \in \mathbb{Z}_{p^a}$ where $p^j \mid u$ and $p^{j+1} \nmid u$ for some non-negative integer j, one can always find a p^a-quasi-inverse by taking $v = (u/p^j)^{-1} \in \mathbb{Z}_{p^{a-j}}$ and naturally lifting v to $\mathbb{Z}_{p^a}$ Therefore, we may let $\mathsf{QuasiInv}(u, p, i)$ be an efficient algorithm outputting a p^i-quasi-inverse element of u and restrict it to output 1 when $p^i \mid u$.

Looking ahead, in our attack, we need to compute 2^{i+1}-quasi-inverse elements for either α_l or $\alpha_l + 2^i$ in each iteration where $\alpha_l = \alpha \mod 2^i$ has been extracted. In a more general case where the prime 2 is replace by q, the attack enumerates q^{i+1}-quasi-inverse elements for $\alpha_l + tq^i$ for every $t \in \{0, \cdots, q-1\}$, which corresponds to guess whether the next digit is t or not. See App. A for more details.

4.2 Attack on HealS and SHealS

The algorithm in Fig. 1 together with Thm. 4.4 provides a recursive approach for recovering α. It requires one random oracle query to recover each bit of α in each iteration.

Algorithm: $\mathsf{Recover}(\mathsf{pp}, \mathsf{sk}_B, \alpha_0)$
Input: pp public parameter of the protocol, sk_B the secret key of Bob,
$\alpha_0 = \alpha \mod 2$
Given: an oracle $\mathcal{O}_\alpha(R_a, S_a, R_{ab}, S_{ab}; E_B, E_{AB}) \to 0/1$ returns 1 if and only if the following equations hold:
$e_{4^a}(R_a, S_a) = e_{4^a}(P_2, Q_2)$,
$\psi'_A(R_a) = [e_1]R_{ab} + [f_1]S_{ab}$,
$\psi'_A(S_a) = [e_2]R_{ab} + [f_2]S_{ab}$,
where ψ'_A is an isogeny from E_B with kernel $\langle [2^a]R_a + [\alpha 2^a]S_a \rangle \in E_B$.
Ensure: Alice's secret key α

Compute $(R_a, S_a, R_{ab}, S_{ab}) \leftarrow (\phi_B(P_2), \phi_B(Q_2), \phi'_B(R_A), \phi'_B(S_A))$ by following the protocol specification using sk_B.
Obtain a from pp.
Obtain $\alpha_l \leftarrow \alpha_0$.
$i = 1$
$j = \perp$ ▷ j will indicate the maximal power of 2 dividing α.
if $\alpha_l = 1$ **then** $j \leftarrow 0$
while $i < a$ **do**
  $\hat{\alpha}_{0l} \leftarrow \mathsf{QuasiInv}(\alpha_l, 2, i+1)$ ▷ $\hat{\alpha}_{0l}(\alpha_l) = 0$ or $2^j \mod 2^{i+1}$
  $\hat{\alpha}_{1l} \leftarrow \mathsf{QuasiInv}(\alpha_l + 2^i, 2, i+1)$ ▷ $\hat{\alpha}_{1l}(\alpha_l + 2^i) = 2^i$ or $2^j \mod 2^{i+1}$
  if $j = \perp$ **then** ▷ Assert $\hat{\alpha}_{0i} = 1$.
    $c \leftarrow \mathcal{O}([1 + 2^{2a-1}]R_a, [\hat{\alpha}_{0l}2^{2a-i-1}]R_a + [1 - 2^{2a-1}]S_a, R_{ab}, S_{ab})$
    $c \leftarrow 1 - c$
    if $c = 0$ **then** $j \leftarrow i$ ▷ Assert 2^j is the maximal power of 2 dividing α.
  else
    $c \leftarrow \mathcal{O}([1 + 2^{2a-i+j-1}]R_a - [\alpha_l 2^{2a-i+j-1}]S_a, [\hat{\alpha}_{0l}2^{2a-i-1}]R_a + [1 - 2^{2a-i+j-1}]S_a, R_{ab}, S_{ab})$
  if $c = 1$ **then** ▷ Assert i-th bit of α is 0.
    $\hat{\alpha}_l \leftarrow \hat{\alpha}_{0l}$
  else ▷ Assert i-th bit of α is 1.
    $\alpha_l \leftarrow \alpha_l + 2^i$
return v

Figure 1: An algorithm to recover the secret α in $\mathsf{sk}_A = (\alpha, e_1, f_1, e_2, f_2)$.

Theorem 4.4. *Assume Alice follows the protocol specification. The algorithm in Fig. 1 returns α in Alice's secret key.*

Proof. We are going to prove the theorem by induction on i for the i-th bit of α where $i < a$. Write $-\alpha = \alpha_l + 2^i\alpha_i + 2^{i+1}\alpha_h \in \mathbb{Z}_{2^a}$ for some $i \in \{1, \ldots, a-1\}$ where $\alpha_l \in \mathbb{Z}_{2^{i-1}}, \alpha_i \in \mathbb{Z}_2, \alpha_h \in \mathbb{Z}_{2^{a-i}}$ represents the bits has been recovered, the next bit to be recovered, and the remaining higher bits respectively. Since we has assumed the correctness of the given least significant bit of α, it suffices to show that given α_l the extraction of α_i, the i-th bit of α, is correct in each iteration of the while-loop of Fig. 1.

Firstly, within each query, the isogeny ψ'_A computed by the oracle is the same because the kernels are all identical:

$$\begin{aligned}\langle [2^a]R_a + [\alpha 2^a]S_a \rangle &= \langle [2^a]([1 + 2^{2a-1}]R_a - [t2^{2a-i-1}]S_a) + [\alpha 2^a]([t'2^{2a-i-1}]R_a + [1 - 2^{2a-1}]S_a) \rangle \\ &= \langle [2^a]([1 + 2^{2a-i+j-1}]R_a - [t2^{2a-i+j-1}]S_a) + [\alpha 2^a]([t'2^{2a-i-1}]R_a + [1 - 2^{2a-i+j-1}]S_a) \rangle,\end{aligned}$$

for any $t, t' \in \mathbb{Z}_{2^a}$ where $i, j \in \mathbb{Z}_a$. Therefore, since R_a, S_a, R_{ab}, S_{ab} are honestly generated, we may assume $e_{4^a}(R_a, S_a) = e_{4^a}(P_2, Q_2)^{3^b}$, $\psi'_A(R_a) = [e_1]R_{ab} + [f_1]S_{ab}$, and $\psi'_A(S_a) = [e_2]R_{ab} + [f_2]S_{ab}$.

Also, every input satisfies Eq. (1). Since $e_{4^a}(R_a, S_a) = e_{4^a}(P_2, Q_2)^{3^b}$, we have for any $\hat{\alpha}_{0l} \in \mathbb{Z}_{2^a}$, and $i, j \in \mathbb{Z}_a$,

$$\begin{aligned}
&e_{4^a}([1 + 2^{2a-1}]R_a - [\alpha_l 2^{2a-i-1}]S_a, [\hat{\alpha}_{0l} 2^{2a-i-1}]R_a + [1 - 2^{2a-1}]S_a) \\
&= e_{4^a}([1 + 2^{2a-i+j-1}]R_a - [\alpha_l 2^{2a-i+j-1}]S_a, [\hat{\alpha}_{0l} 2^{2a-i-1}]R_a + [1 - 2^{2a-i+j-1}]S_a) \\
&= e_{4^a}(R_a, S_a) \\
&= e_{4^a}(P_2, Q_2)^{3^b}.
\end{aligned}$$

To prove the correctness of the extraction of α_i, we claim that Eqs. (2) and (3) are both satisfied if and only if α_i is 1 in the if-loop of $j = \perp$ or is 0 in the if-loop of $j \neq \perp$. We therefore consider two cases.

Case1: the if-loop of $j = \perp$. The condition is equivalent to $\alpha_l = 0$ which means $\alpha = 0 \mod 2^i$. Recall $\psi'_A(R_a) = [e_1]R_{ab} + [f_1]S_{ab}$, and $\psi'_A(S_a) = [e_2]R_{ab} + [f_2]S_{ab}$. For Eq. (2), we have

$$\begin{aligned}
&\psi'_A([1 + 2^{2a-1}]R_a) - [e_1]R_{ab} - [f_1]S_{ab} \\
&= [(1 + 2^{2a-1})e_1]R_{ab} + [(1 + 2^{2a-1})f_1]S_{ab} - [e_1]R_{ab} - [f_1]S_{ab} \\
&= [2^{2a-1}e_1]R_{ab} + [2^{2a-1}f_1]S_{ab} \\
&= [-\alpha 2^{2a-1}e_2]R_{ab} + [-\alpha 2^{2a-1}f_2]S_{ab} \\
&= \mathbf{O}.
\end{aligned}$$

That is, Eq. (2) will always hold. Remark the third equation comes from Lem. 2.3 and the fact that i is less than a. The fourth equation comes from the fact that $\alpha = 0 \mod 2^i$ and $i \geq 1$ and $\{R_{ab}, S_{ab}\}$ is a basis for $E_{AB}[4^a]$.

Also, since $\alpha_l = 0$, $\hat{\alpha}_{0l}$ is 1 by the specification of $\mathsf{QuasiInv}$. Recall $\psi'_A(R_a) = [e_1]R_{ab} + [f_1]S_{ab}$, and $\psi'_A(S_a) = [e_2]R_{ab} + [f_2]S_{ab}$. For Eq. (3), we have

$$\begin{aligned}
&\psi'_A([\hat{\alpha}_{0l} 2^{2a-i-1}]R_a + [1 - 2^{2a-1}]S_a) - [e_2]R_{ab} - [f_2]S_{ab} \\
&= [2^{2a-i-1}e_1 + (1 - 2^{2a-1})e_2]R_{ab} + [2^{2a-i-1}f_1 + (1 - 2^{2a-1})f_2]S_{ab} - [e_1]R_{ab} - [f_1]S_{ab} \\
&= [2^{2a-i-1}e_1 - 2^{2a-1}e_2]R_{ab} + [2^{2a-i-1}f_1 - 2^{2a-1}f_2]S_{ab} \\
&= [-\alpha 2^{2a-i-1}e_2 - 2^{2a-1}e_2]R_{ab} + [-\alpha 2^{2a-i-1}f_2 - 2^{2a-1}f_2]S_{ab} \\
&= [\alpha_i 2^{2a-1} - 2^{2a-1}][e_2]R_{ab} + [\alpha_i 2^{2a-1} - 2^{2a-1}][f_2]S_{ab}.
\end{aligned}$$

Similarly, the third equation comes from Lem. 2.3 and the fact that i is less than a. The fourth equation comes from the fact that $\alpha = 0 \mod 2^i$ and $\{R_{ab}, S_{ab}\}$ is a basis for $E_{AB}[4^a]$. By Lem. 3.1, e_2 and f_2 cannot be both even so that at least one of $[2^{2a-1}e_2]R_{ab}$ and $[2^{2a-1}f_2]S_{ab}$ is of order 2. Eq. (3) holds if and only if α_i is 1.

Therefore, by combining conditions of Eqs. (1) to (3), in the if-loop of $j = \perp$, the oracle outputs $c = 1$ if and only if $\alpha_i = 1$.

Case2: the if-loop of $j \neq \perp$. The condition is equivalent to 2^j is the maxmal power of 2 dividing α. Recall $\psi'_A(R_a) = [e_1]R_{ab} + [f_1]S_{ab}$, and $\psi'_A(S_a) = [e_2]R_{ab} + [f_2]S_{ab}$. For Eq. (2), we have

$$\begin{aligned}
&\psi'_A([1 + 2^{2a-i+j-1}]R_a - [\alpha_l 2^{2a-i+j-1}]S_a) - [e_1]R_{ab} - [f_1]S_{ab} \\
&= [(1 + 2^{2a-i+j-1})e_1 - \alpha_l 2^{2a-i+j-1}e_2]R_{ab} + [(1 + 2^{2a-i+j-1})f_1 - \alpha_l 2^{2a-i+j-1}f_2]S_{ab} - [e_1]R_{ab} - [f_1]S_{ab} \\
&= [2^{2a-i+j-1}e_1 - \alpha_l 2^{2a-i+j-1}e_2]R_{ab} + [2^{2a-i+j-1}f_1 - \alpha_l 2^{2a-i+j-1}f_2]S_{ab} \\
&= [-\alpha 2^{2a-i+j-1} - \alpha_l 2^{2a-i+j-1}][e_2]R_{ab} + [-\alpha 2^{2a-i+j-1} - \alpha_l 2^{2a-i+j-1}][f_2]S_{ab}
\end{aligned}$$

Remark the third equation comes from Lem. 2.3 and the fact that $i-j$ is less than a. When $j \geq 1$, we have $-\alpha 2^{2a-i+j-1} - \alpha_l 2^{2a-i+j-1} = 0 \mod 4^a$. In this case, the oracle will return 1. When $j = 0$, we have $-\alpha 2^{2a-i+j-1} - \alpha_l 2^{2a-i+j-1} = \alpha_i 2^{2a-1} \mod 4^a$. In this case, the oracle will return 1 if and only if $\alpha_i = 0$. Therefore, Eq. (2) hold if and only if α is even or $\alpha_i = 0$.

Also, for Eq. (3), we have $\hat{\alpha}$

$$
\begin{aligned}
&\psi'_A([\hat{\alpha}_{0l}2^{2a-i-1}]R_a + [1-2^{2a-i+j-1}]S_a) - [e_2]R_{ab} - [f_2]S_{ab} \\
&= [\hat{\alpha}_{0l}2^{2a-i-1}e_1 + (1-2^{2a-i+j-1})e_2]R_{ab} + [\hat{\alpha}_{0l}2^{2a-i-1}f_1 + (1-2^{2a-i+j-1})f_2]S_{ab} - [e_1]R_{ab} - [f_1]S_{ab} \\
&= [\hat{\alpha}_{0l}2^{2a-i-1}e_1 - 2^{2a-i+j-1}e_2]R_{ab} + [\hat{\alpha}_{0l}2^{2a-i-1}f_1 - 2^{2a-i+j-1}f_2]S_{ab} \\
&= [-\alpha\hat{\alpha}_{0l}2^{2a-i-1}e_2 - 2^{2a-i+j-1}e_2]R_{ab} + [-\alpha\hat{\alpha}_{0l}2^{2a-i-1}f_2 - 2^{2a-i+j-1}f_2]S_{ab} \\
&= [\alpha_i\hat{\alpha}_{0l}2^{2a-i+j}][e_2]R_{ab} + [\alpha_i\hat{\alpha}_{0l}2^{2a-i+j}][f_2]S_{ab}.
\end{aligned}
$$

Similarly, the third equation comes from Lem. 2.3 and the fact that i is less than a. The fourth equation comes from the fact that $\hat{\alpha}_{0l}$ is a 2^{i+1}-quasi-inverse element for α_l. That is, $\hat{\alpha}_{0l}\alpha_l = 2^j \mod 2^{i+1}$. Therefore, $-\alpha\hat{\alpha}_{0l}2^{2a-i-1} = 2^{2a-i+j-1} + \alpha_i\hat{\alpha}_{0l}2^{2a-i+j} \mod 4^a$ where $i-j > 0$. By Lem. 3.1, e_2 and f_2 cannot be both even so that at least one of $[2^{2a-1}e_2]R_{ab}$ and $[2^{2a-1}f_2]S_{ab}$ is not $\mathbf{O}$. By Prop. 4.3, $\hat{\alpha}_{0l}$ is a unit. Therefore, in this case we know Eq. (3) holds if and only if the next bit α_i to extracted is 0.

Therefore, by combining conditions of Eqs. (1) to (3), in the if-loop of $j \neq \perp$, the oracle outputs $c = 1$ if and only if $\alpha_i = 0$.

□

5 Summary

It takes one oracle query to recover each bit. In the appendix, we consider a more generic situation for HealSIDH where 2 can be replaced by any distinct prime number q. The algorithm takes $a(q-1)$ oracle queries to fully recover Alice's secret key $\alpha \in \mathbb{Z}_{q^a}$.

Acknowledgement

This project is supported by the Ministry for Business, Innovation and Employment in New Zealand.

References

[ACC+17] Reza Azarderakhsh, Matthew Campagna, Craig Costello, LD Feo, Basil Hess, Amir Jalali, David Jao, Brian Koziel, Brian LaMacchia, Patrick Longa, et al. Supersingular isogeny key encapsulation. *submission to the NIST post-quantum standardization project*, 152:154–155, 2017.

[AJL17] Reza Azarderakhsh, David Jao, and Christopher Leonardi. Post-quantum static-static key agreement using multiple protocol instances. In Carlisle Adams and Jan Camenisch, editors, *SAC 2017*, volume 10719 of *LNCS*, pages 45–63. Springer, Heidelberg, August 2017.

[BDK+21] Ward Beullens, Samuel Dobson, Shuichi Katsumata, Yi-Fu Lai, and Federico Pintore. Group signatures and more from isogenies and lattices: Generic, simple, and efficient. Cryptology ePrint Archive, Report 2021/1366, 2021. https://eprint.iacr.org/2021/1366.

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

[FP21] Tako Boris Fouotsa and Christophe Petit. SHealS and HealS: isogeny-based PKEs from akey validation method for SIDH. Cryptology ePrint Archive, Report 2021/1596, 2021. https://eprint.iacr.org/2021/1596.

[GPST16] Steven D. Galbraith, Christophe Petit, Barak Shani, and Yan Bo Ti. On the security of supersingular isogeny cryptosystems. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *ASIACRYPT 2016, Part I*, volume 10031 of *LNCS*, pages 63–91. Springer, Heidelberg, December 2016.

[JD11] David Jao and Luca De Feo. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. In Bo-Yin Yang, editor, *Post-Quantum Cryptography - 4th International Workshop, PQCrypto 2011*, pages 19–34. Springer, Heidelberg, November / December 2011.

[Kup05] Greg Kuperberg. A subexponential-time quantum algorithm for the dihedral hidden subgroup problem. *SIAM Journal on Computing*, 35(1):170–188, 2005.

[LD21] Yi-Fu Lai and Samuel Dobson. Collusion resistant revocable ring signatures and group signatures from hard homogeneous spaces. Cryptology ePrint Archive, Report 2021/1365, 2021. https://eprint.iacr.org/2021/1365.

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

[MOT20] 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, *ASIACRYPT 2020, Part II*, volume 12492 of *LNCS*, pages 551–580. Springer, Heidelberg, December 2020.

[Sil09] Joseph H Silverman. *The arithmetic of elliptic curves*, volume 106. Springer, 2009.

[UJ20] David Urbanik and David Jao. New techniques for sidh-based nike. *Journal of Mathematical Cryptology*, 14(1):120–128, 2020.

A A Generalized Attack

This section presents a generalized result. We consider a more generic condition where Alice uses q^{2a} torsion subgroup instead of 2^{2a}. The final algorithm takes $a(q-1)$ oracle queries to fully recover Alice's secret key $\alpha \in \mathbb{Z}_{q^a}$.

To be more specific, the public key parameter $\mathsf{pp} = (E_0, P_q, Q_q, P_{q'}, Q_{q'}, p, q, q')$ where p, q, q' are distinct primes, $p = fq^{2a}q'^{2b} - 1$, $q^a \approx q'^b$, and $\{P_q, Q_q\}$ and $\{P_{q'}, Q_{q'}\}$ are bases for $E_0[q^{2a}]$ and $E_0[q'^{2b}]$, resp. Let $[q^a]P_q = P_A$ and $[q^a]Q_q = Q_A$. Alice samples a secret α uniformly at random from $\mathbb{Z}_{q^a}$, computes $\phi_A : E_0 \to E_A = E_0/\langle P_A + [\alpha]Q_A\rangle$ and representing $\phi_A(P_q) = [e_1]R_A + [f_1]S_A$ and $\phi_A(Q_q) = [e_2]R_A + [f_2]S_A$ where $\{R_A, S_A\}$ is a canonical basis for $E_A[q^{2a}]$. Alice's secret key is $\mathsf{sk}_A = (\alpha, e_1, f_1, e_2, f_2)$ and public key is $(E_A, \phi_A(P_{q'}), \phi_A(Q_{q'}))$.

We start we following three simple facts similar to Prop. 2.2 and Lems. 2.3 and 3.1.

Proposition A.1. *If Bob honestly will generate R_a, S_a, R_{ab}, S_{ab} by $R_a = \phi_B(P_2)$, $S_a = \phi'_B(Q_2)$, $R_{ab} = \phi'_B(R_A)$ and $S_{ab} = \phi_B(S_A)$, then $\{R_{ab}, S_{ab}\}$ is a basis of $E_{AB}[q^{2a}]$ and $\{R_a, S_a\}$ is a basis of $E_B[q^{2a}]$.*

Proof. Since $[q^{2a}]R_a = \phi_B([q^{2a}]P_2) = \mathbf{O}$ and $[q^{2a}]S_a = \phi'_B([q^{2a}]Q_2) = \mathbf{O}$, both R_a and S_a are in $E_B[q^{2a}]$. Due to $e_{q^{2a}}(R_a, S_a) = e_{q^{2a}}(P_2, Q_2)^{q'^b}$, we know $e_{q^{2a}}(R_a, S_a)$ is a primitive q^{2a}-th root of unity. Similarly, Since $[q^{2a}]R_{ab} = \phi'_B([q^{2a}]R_A) = \mathbf{O}$ and $[q^{2a}]S_{ab} = \phi_B([q^{2a}]S_A) = \mathbf{O}$, both R_{ab} and S_{ab} are in $E_{AB}[q^{2a}]$. Due

to $e_{q^{2a}}(R_{ab}, S_{ab}) = e_{q^{2a}}(R_A, S_A)^{q'^b}$, we know $e_{q^{2a}}(R_{ab}, S_{ab})$ is a primitive q^{2a}-th root of unity. Therefore, the result follows. □

Lemma A.2. *Let e_1, e_2, f_1, f_2 defined as above and $\alpha \in \mathbb{Z}_{q^a}$ be the secret key of Alice such that $\ker(\phi_A) = \langle [q^a]P_2 + [\alpha q^a]Q_2 \rangle$. If Alice follows the protocol specification, then $e_1 + \alpha e_2 = f_1 + \alpha f_2 = 0 \mod q^a$.*

Proof. Given $\phi_A(P_2) = [e_1]R_A + [f_1]S_A$ and $\phi_A(Q_2) = [e_2]R_A + [f_2]S_A$, we have $\mathbf{O} = \phi_A([q^a]P_2 + [\alpha q^a]Q_2) = [q^a e_1 + \alpha q^a e_2]R_A + [q^a f_1 + \alpha q^a f_2]S_A = [e_1 + \alpha e_2]([q^a]R_a) + [f_1 + \alpha f_2]([q^a]S_A)$.

Recall that $\{[q^a]R_A, [q^a]S_A\}$ is a basis for $E_A[q^a]$ due to $\{R_A, S_A\}$ being a basis for $E_A[q^{2a}]$. Therefore, $e_1 + \alpha e_2 = f_1 + \alpha f_2 = 0 \mod q^a$. □

Lemma A.3. *If Alice produces $\phi_A(P_q)$ and $\phi_A(Q_q)$ honestly, then f_1 and f_2 cannot be both divisible by q.*

Proof. Suppose for the purpose of contradiction that both e_2 and f_2 are even. Then, $[q^{2a-1}]\phi_A(Q_q) = \mathbf{O}$, which implies $\ker(\phi_A) = \langle P_q + [\alpha]Q_q \rangle$ contains $[q^{2a-1}]Q_q$. That is, $[k]P_q + [k\alpha]Q_2 = [q^{2a-1}]Q_2$ for some $k \in \mathbb{Z}_{q^a}$, so $k = 0$. This contradicts the fact that $\{P_q, Q_q\}$ is a basis for $E_0[q^{2a}]$. The result follows. □

The algorithm in Fig. 2 together with Thm. A.4 provides a recursive approach for recovering α. It requires one random oracle quries to recover each bit of α in each iteration.

Theorem A.4. *Assume Alice follows the protocol specification. The algorithm in Fig. 2 returns α in Alice's secret key.*

Proof. We are going to prove the theorem by induction on i for the i-th bit of α where $i < a$. Write $-\alpha = \alpha_l + q^i\alpha_i + q^{i+1}\alpha_h \in \mathbb{Z}_{q^a}$ for some $i \in \{0, \ldots, a-1\}$ where $\alpha_l \in \mathbb{Z}_{q^{i-1}}, \alpha_i \in \mathbb{Z}_2, \alpha_h \in \mathbb{Z}_{q^{a-i}}$ represents the bits has been recovered, the next bit to be recovered, and the remaining higher bits respectively.

Firstly, within each query, the isogeny ψ'_A computed by the oracle is the same, because the kernels are all identical:

$$\begin{aligned}\langle [q^a]R_a + [\alpha q^a]S_a \rangle &= \langle [q^a]([1 + q^{2a-1}]R_a - [t'q^{2a-i-1}]S_a) + [\alpha q^a]([tq^{2a-i-1}]R_a + [1 - q^{2a-1}]S_a) \rangle \\ &= \langle [q^a]([1 + q^{2a-i+j-1}]R_a - [t'q^{2a-i+j-1}]S_a) + [\alpha q^a]([tq^{2a-i-1}]R_a + [1 - q^{2a-i+j-1}]S_a) \rangle,\end{aligned}$$

where $i, j \in \mathbb{Z}_a$ and arbitrary $t, t' \in \mathbb{Z}_{q^a}$. Therefore, since R_a, S_a, R_{ab}, S_{ab} are honestly generated, we may assume $e_{q^a}(R_a, S_a) = e_{q^a}(P_2, Q_2)^{q'^b}$, $\psi'_A(R_a) = [e_1]R_{ab} + [f_1]S_{ab}$, and $\psi'_A(S_a) = [e_2]R_{ab} + [f_2]S_{ab}$.

Also, every input satisfies Eq. (1). Since $e_{q^a}(R_a, S_a) = e_{q^a}(P_2, Q_2)^{q'^b}$, we have for any $t, t' \in \mathbb{Z}_{q^a}$, and $i, j \in \mathbb{Z}_a$,

$$\begin{aligned}&e_{q^{2a}}([1 + q^{2a-1}]R_a - [t'q^{2a-i-1}]S_a, [tq^{2a-i-1}]R_a + [1 - q^{2a-1}]S_a) \\ &= e_{q^{2a}}([1 + q^{2a-i+j-1}]R_a - [t'q^{2a-i+j-1}]S_a, [tq^{2a-i-1}]R_a + [1 - q^{2a-i+j-1}]S_a) \\ &= e_{q^{2a}}(R_a, S_a) \\ &= e_{q^{2a}}(P_2, Q_2)^{q'^b}.\end{aligned}$$

For the case $i = 0$ of induction, we have to show the correctness of the extraction of α_0, the least significant bit of $-\alpha$. We restrict our attention to the if-loop of the condition $i = 0$. Recall $\psi'_A(R_a) = [e_1]R_{ab} + [f_1]S_{ab}$, and $\psi'_A(S_a) = [e_2]R_{ab} + [f_2]S_{ab}$. For Eq. (2) $t \in \mathbb{Z}_q$, we have

$$\begin{aligned}&\psi'_A([1 + q^{2a-1}]R_a - [tq^{2a-1}]S_a) - [e_1]R_{ab} - [f_1]S_{ab} \\ &= [(1 + q^{2a-1})e_1 - tq^{2a-1}e_2]R_{ab} + [(1 + q^{2a-1})f_1 - tq^{2a-1}f_2]S_{ab} - [e_1]R_{ab} - [f_1]S_{ab} \\ &= [q^{2a-1}e_1 - tq^{2a-1}e_2]R_{ab} + [q^{2a-1}f_1 - tq^{2a-1}f_2]S_{ab} \\ &= [-\alpha q^{2a-1}e_2 - tq^{2a-1}e_2]R_{ab} + [-\alpha q^{2a-1}f_2 - tq^{2a-1}f_2]S_{ab} \\ &= [\alpha_0 q^{2a-1}e_2 - tq^{2a-1}e_2]R_{ab} + [\alpha_0 q^{2a-1}f_2 - tq^{2a-1}f_2]S_{ab}\end{aligned}$$

Algorithm: $\mathsf{Recover}(\mathsf{pp}, \mathsf{sk}_B)$
Input: pp public parameter of the protocol, sk_B the secret key of Bob,
Given: an oracle $\mathcal{O}_\alpha(R_a, S_a, R_{ab}, S_{ab}; E_B, E_{AB}) \to 0/1$ returns 1 if and only if the following equations hold:
$e_{q^{2a}}(R_a, S_a) = e_{q^{2a}}(P_q, Q_q)$,
$\psi'_A(R_a) = [e_1]R_{ab} + [f_1]S_{ab}$,
$\psi'_A(S_a) = [e_2]R_{ab} + [f_2]S_{ab}$,
where ψ'_A is an isogeny from E_B with kernel $\langle [q^a]R_a + [\alpha q^a]S_a \rangle \in E_B$.
Ensure: Alice's secret key α

Obtain $(R_a, S_a, R_{ab}, S_{ab}) \leftarrow (\phi_B(P_q), \phi_B(Q_q), \phi'_B(R_A), \phi'_B(S_A))$ by following the protocol specification using sk_B.
Obtain a from pp.
$i = 0$
$j = \perp$
while $i < a$ **do**
  $c = 0$
  $t = q$
  for $t \in \{0, \cdots, q-1\}$ **do**
    $\hat{\alpha}_{tl} \leftarrow \mathsf{QuasiInv}(\alpha_l + tq^i, q, i+1)$
  if $i = 0$ **then** ▷ Extract α_0.
    while $c = 0$ or $t > 0$ **do**
      $t \mathrel{-}= 1$
      $c \leftarrow \mathcal{O}([1 + q^{2a-1}]R_a - [tq^{2a-1}]S_a, [\hat{\alpha}_{tl}q^{2a-1}]R_a + [1 - q^{2a-1}]S_a, R_{ab}, S_{ab})$
    $\alpha_l \leftarrow t$
    $i \mathrel{+}= 1$
    if $t \neq 0$ **then** $j \leftarrow i$ ▷ Assert q is the maximal power of q dividing α.
  if $j = \perp$ **then** ▷ Assert $\hat{\alpha}_{tl}t = 1$ or $0 \mod q$.
    while $c = 0$ or $t > 0$ **do**
      $t \mathrel{-}= 1$
      $c \leftarrow \mathcal{O}([1 + q^{2a-1}]R_a, [\hat{\alpha}_{tl}q^{2a-i-1}]R_a + [1 - q^{2a-1}]S_a, R_{ab}, S_{ab})$
    $\alpha_l \leftarrow \alpha_l + tq^i$ ▷ Assert i-th bit of α is t.
    if $t \neq 0$ **then** $j \leftarrow i$ ▷ Assert q^j is the maximal power of q dividing α.
  else ▷ Assert $\hat{\alpha}_{tl}(\alpha_l + tq^i) = q^j \mod q^{i+1}$.
    while $c = 0$ or $t > 0$ **do**
      $t \mathrel{-}= 1$
      $c \leftarrow \mathcal{O}([1 + q^{2a-i+j-1}]R_a - [(\alpha_l + tq^i)q^{2a-i+j-1}]S_a, [\hat{\alpha}_{l\tilde{\alpha}_i}q^{2a-i-1}]R_a + [1 - q^{2a-i+j-1}]S_a, R_{ab}, S_{ab})$
    $\alpha_l \leftarrow \alpha_l + tq^i$ ▷ i-th digit of α is t.
  $i \mathrel{+}= 1$
return $-v \mod q^a$

Figure 2: A general algorithm to recover the secret α.

That is, Eq. (2) will always hold. Remark the third equation comes from Lem. A.2. Therefore, the condition of Eq. (2) is satisfied if and only if $t = \alpha_0$.

Similarly, for Eq. (3), we have

$$\begin{aligned}
&\psi'_A([\hat{\alpha}_{tl}q^{2a-i-1}]R_a + [1-q^{2a-1}]S_a) - [e_2]R_{ab} - [f_2]S_{ab} \\
&= [\hat{\alpha}_{tl}q^{2a-1}e_1 + (1-q^{2a-1})e_2]R_{ab} + [\hat{\alpha}_{tl}q^{2a-1}f_1 + (1-q^{2a-1})f_2]S_{ab} - [e_1]R_{ab} - [f_1]S_{ab} \\
&= [\hat{\alpha}_{tl}q^{2a-1}e_1 - q^{2a-1}e_2]R_{ab} + [\hat{\alpha}_{tl}q^{2a-1}f_1 - q^{2a-1}f_2]S_{ab} \\
&= [-\alpha\hat{\alpha}_{tl}q^{2a-1}e_2 - q^{2a-1}e_2]R_{ab} + [-\alpha\hat{\alpha}_{tl}q^{2a-1}f_2 - q^{2a-1}f_2]S_{ab} \\
&= [\alpha_0\hat{\alpha}_{tl}q^{2a-1} - q^{2a-1}][e_2]R_{ab} + [\alpha_0\hat{\alpha}_{tl}q^{2a-1} - q^{2a-1}][f_2]S_{ab}.
\end{aligned}$$

That is, Eq. (3) will always hold. Remark the third equation comes from Lem. A.2. Therefore, the condition of Eq. (3) is satisfied if and only if $\alpha_0\hat{\alpha}_{tl} = 1 \mod q$. Equivalently, $t = \alpha_0$, because $t\hat{\alpha}_{tl} = 1 \mod q$. If $\alpha_0\hat{\alpha}_{tl} \neq 1 \mod q$ for all $t \in \{1, \cdots, q-1\}$, then $\alpha_0 = 0$. Therefore, by combining conditions of Eqs. (1) to (3), the extraction of α_0 is correct.

It suffices to show that given α_l the extraction of α_i, the i-th bit of $-\alpha \mod q^a$ for $i \geq 1$, is correct in each iteration of the while-loop of Fig. 2. To prove the correctness of the extraction of α_i, in either the if-loop of $j = \perp$ or the else-loop ($j \neq \perp$), we claim that Eqs. (2) and (3) are both satisfied if and only if the output of the oracle is $c = 1$ for $t \in \{1, \cdots, q-1\}$ used in the loop if and only if $\alpha_i = t$ for some $t \in \{1, \cdots, q-1\}$. We therefore consider two cases.

Case1: the if-loop of $j = \perp$. The condition is equivalent to $\alpha_l = 0$ which means $-\alpha = 0 \mod q^i$. Recall $\psi'_A(R_a) = [e_1]R_{ab} + [f_1]S_{ab}$, and $\psi'_A(S_a) = [e_2]R_{ab} + [f_2]S_{ab}$. For Eq. (2), we have

$$\begin{aligned}
&\psi'_A([1+q^{2a-1}]R_a) - [e_1]R_{ab} - [f_1]S_{ab} \\
&= [(1+q^{2a-1})e_1]R_{ab} + [(1+q^{2a-1})f_1]S_{ab} - [e_1]R_{ab} - [f_1]S_{ab} \\
&= [q^{2a-1}e_1]R_{ab} + [q^{2a-1}f_1]S_{ab} \\
&= [-\alpha q^{2a-1}e_2]R_{ab} + [-\alpha q^{2a-1}f_2]S_{ab} \\
&= \mathbf{O}.
\end{aligned}$$

That is, Eq. (2) will always hold. Remark the third equation comes from Lem. A.2 and the fact that i is less than a. The fourth equation comes from the fact that $\alpha = 0 \mod q^i$ for $i \geq 1$ and $\{R_{ab}, S_{ab}\}$ is a basis for $E_{AB}[q^{2a}]$.

Also, since $\alpha_l = 0$, $\hat{\alpha}_{tl}t = 1 \mod q$. Recall $\psi'_A(R_a) = [e_1]R_{ab} + [f_1]S_{ab}$, and $\psi'_A(S_a) = [e_2]R_{ab} + [f_2]S_{ab}$. For Eq. (3), we have

$$\begin{aligned}
&\psi'_A([\hat{\alpha}_{tl}q^{2a-i-1}]R_a + [1-q^{2a-1}]S_a) - [e_2]R_{ab} - [f_2]S_{ab} \\
&= [\hat{\alpha}_{tl}q^{2a-i-1}e_1 + (1-q^{2a-1})e_2]R_{ab} + [\hat{\alpha}_{tl}q^{2a-i-1}f_1 + (1-q^{2a-1})f_2]S_{ab} - [e_1]R_{ab} - [f_1]S_{ab} \\
&= [\hat{\alpha}_{tl}q^{2a-i-1}e_1 - q^{2a-1}e_2]R_{ab} + [\hat{\alpha}_{tl}q^{2a-i-1}f_1 - q^{2a-1}f_2]S_{ab} \\
&= [-\alpha\hat{\alpha}_{tl}q^{2a-i-1}e_2 - q^{2a-1}e_2]R_{ab} + [-\alpha\hat{\alpha}_{tl}q^{2a-i-1}f_2 - q^{2a-1}f_2]S_{ab} \\
&= [\alpha_i\hat{\alpha}_{tl}q^{2a-1} - q^{2a-1}][e_2]R_{ab} + [\alpha_i\hat{\alpha}_{tl}q^{2a-1} - q^{2a-1}][f_2]S_{ab}.
\end{aligned}$$

Similarly, the third equation comes from Lem. A.2 and the fact that i is less than a. The fourth equation comes from the fact that $\alpha = 0 \mod q^i$ and $\{R_{ab}, S_{ab}\}$ is a basis for $E_{AB}[q^{2a}]$. By Lem. A.3, e_2 and f_2 cannot be both even so that at least one of $[q^{2a-1}e_2]R_{ab}$ and $[q^{2a-1}f_2]S_{ab}$ is of order q. Eq. (3) holds if and only if the next bit to extracted α_i is $t = \hat{\alpha}_{tl}^{-1} \mod q$. If $\alpha_i\hat{\alpha}_{tl} \neq 1 \mod q$ for all $t \in \{1, \cdots, q-1\}$, then $\alpha_i = 0$. Therefore, by combining conditions of Eqs. (1) to (3), in the if-loop of $j = \perp$, the oracle outputs

$c = 1$ for $t \in \{1, q-1\}$ used in the loop if and only if $\alpha_i = t$. Moreover, if all outputs of the oracle in the loop is 0, then $\alpha_i = 0$.

Case2: the if-loop of $j \neq \perp$. The condition is equivalent to q^j is the maxmal power of q dividing α. Recall $\psi'_A(R_a) = [e_1]R_{ab} + [f_1]S_{ab}$, and $\psi'_A(S_a) = [e_2]R_{ab} + [f_2]S_{ab}$. For Eq. (2), we have

$$\begin{aligned}
&\psi'_A([1+q^{2a-i+j-1}]R_a - [(\alpha_l + tq^i)q^{2a-i+j-1}]S_a) - [e_1]R_{ab} - [f_1]S_{ab} \\
&= [(1+q^{2a-i+j-1})e_1 - (\alpha_l + tq^i)q^{2a-i+j-1}e_2]R_{ab} + [(1+q^{2a-i+j-1})f_1 - (\alpha_l + tq^i)q^{2a-i+j-1}f_2]S_{ab} - [e_1]R_{ab} - [f_1]S_{ab} \\
&= [q^{2a-i+j-1}e_1 - (\alpha_l + tq^i)q^{2a-i+j-1}e_2]R_{ab} + [q^{2a-i+j-1}f_1 - (\alpha_l + tq^i)q^{2a-i+j-1}f_2]S_{ab} \\
&= [-\alpha q^{2a-i+j-1} - (\alpha_l + tq^i)q^{2a-i+j-1}][e_2]R_{ab} + [-\alpha q^{2a-i+j-1} - (\alpha_l + tq^i)q^{2a-i+j-1}][f_2]S_{ab} \\
&= [(\alpha_i - t)q^{2a-1}][e_2]R_{ab} + [(\alpha_i - t)q^{2a-1}][f_2]S_{ab}
\end{aligned}$$

Remark the third equation comes from Lem. A.2 and the fact that $i - j$ is less than a. If $j \geq 1$, we always have $-\alpha q^{2a-i+j-1} - (\alpha_l + tq^i)q^{2a-i+j-1} = 0 \mod q^{2a}$. In this case, the oracle will return 1. If $j = 0$, we have $-\alpha q^{2a-i+j-1} - (\alpha_l + tq^i)q^{2a-i+j-1} = (\alpha_i - t)q^{2a-1} \mod q^{2a}$. In this case, the oracle will return 1 if and only if $\alpha_i = t$. If $\alpha_i = 0$, then the oracle will always return 0. Therefore, Eq. (2) hold if and only if $q \mid \alpha$ or $\alpha_i = t$ for some $t \in \{1, \cdots, q-1\}$.

Also, for Eq. (3), we have $\hat{\alpha}$

$$\begin{aligned}
&\psi'_A([\hat{\alpha}_{tl}q^{2a-i-1}]R_a + [1-q^{2a-i+j-1}]S_a) - [e_2]R_{ab} - [f_2]S_{ab} \\
&= [\hat{\alpha}_{tl}q^{2a-i-1}e_1 + (1-q^{2a-i+j-1})e_2]R_{ab} + [\hat{\alpha}_{tl}q^{2a-i-1}f_1 + (1-q^{2a-i+j-1})f_2]S_{ab} - [e_1]R_{ab} - [f_1]S_{ab} \\
&= [\hat{\alpha}_{tl}q^{2a-i-1}e_1 - q^{2a-i+j-1}e_2]R_{ab} + [\hat{\alpha}_{tl}q^{2a-i-1}f_1 - q^{2a-i+j-1}f_2]S_{ab} \\
&= [-\alpha\hat{\alpha}_{tl}q^{2a-i-1}e_2 - q^{2a-i+j-1}e_2]R_{ab} + [-\alpha\hat{\alpha}_{tl}q^{2a-i-1}f_2 - q^{2a-i+j-1}f_2]S_{ab} \\
&= [(\alpha_i - t)\hat{\alpha}_{tl}q^{2a-i+j}][e_2]R_{ab} + [(\alpha_i - t)\hat{\alpha}_{tl}q^{2a-i+j}][f_2]S_{ab}.
\end{aligned}$$

Similarly, the third equation comes from Lem. A.2 and the fact that i is less than a. The fourth equation comes from the fact that $\hat{\alpha}_{tl}$ is a p^{i+1}-quasi-inverse element for $\alpha_l + tq^i$. That is, $\hat{\alpha}_{tl}(\alpha_l + tq^i) = q^j \mod q^{i+1}$. Therefore, $-\alpha\hat{\alpha}_{tl}q^{2a-i-1} = q^{2a-i+j-1} + (\alpha_i - t)\hat{\alpha}_{tl}q^{2a-i+j} \mod q^{2a}$ where $i - j \geq 1$. By Lem. A.3, e_2 and f_2 cannot be both divisible by q so that at least one of $[q^{2a-1}e_2]R_{ab}$ and $[q^{2a-1}f_2]S_{ab}$ is not $\mathbf{O}$. By Prop. 4.3, $\hat{\alpha}_{tl}$ is a unit. Therefore, in this case we know Eq. (3) holds if and only if $-\alpha\hat{\alpha}_{tl} = 1 \mod q^{i+1}$ if and only if $\alpha_i = t$.

Therefore, by combining conditions of Eqs. (1) to (3), in the if-loop of $j \neq \perp$, the oracle outputs $c = 1$ for $t \in \{1, \cdots, q-1\}$ used in the loop if and only if $\alpha_i = t$. Moreover, if all outputs of the oracle in the loop is 0, then $\alpha_i = 0$.

□