INKE: Isogeny-Based PKE Using Intermediate Curves

Hyeonhak Kim[1], Won Kim[1], Changmin Lee[1], Suhri Kim[3], Seokhie Hong[2], and Sangjin Lee[1]

[1] School of Cybersecurity, Korea University, Seoul 02841, South Korea
[2] SmartM2M, South Korea
[3] School of Mathematics, Statistics and Data Science, Sungshin Women's University, South Korea

Abstract. POKÉ (POint-based Key Exchange), proposed by Basso and Maino at Eurocrypt 2025, is currently the fastest known isogeny-based public-key encryption scheme. Although POKÉ is secure against currently known key-recovery attacks, there is no known reduction from key-recovery security to IND-CPA security.
In this work, we propose INKE, a variant of POKÉ that replaces torsion points in the encryption process with intermediate elliptic curves. This modification enables a quantum reduction from key-recovery security to IND-CPA security in the algebraic isogeny model (AIM), while maintaining the practical performance.
Although INKE is overall slower than POKÉ and has larger public-key and ciphertext sizes, it remains more efficient than other group-action-based key exchange protocols such as CSIDH and CORAL that admit reductions from key-recovery security to shared-secret security in algebraic group action model (AGAM). To illustrate the practical overhead of INKE compared to POKÉ, we provide an optimized C implementation together with detailed benchmark comparisons at each security level.

Keywords: Post Quantum Cryptography, Isogeny, SIDH, Higher Dimension, POKÉ, Algebraic Isogeny Model

1 Introduction

Isogeny-based cryptography has attracted significant attention as a promising approach to post-quantum cryptography, particularly because of its potential for compact public keys, ciphertexts, and signatures. The SIDH protocol [24] introduced a practical key exchange mechanism based on supersingular elliptic curves over $\mathbb{F}_{p^2}$, and its KEM variant SIKE was selected as an alternate candidate in the NIST post-quantum cryptography standardization process.

A distinctive feature of SIDH is its non-commutative structure. In order to derive a common shared secret, each party evaluates its secret isogeny on the other party's public torsion basis and publishes the resulting images. Castryck and Decru [9] showed that these torsion-point images, together with Kani's lemma, can

be used to recover the secret isogeny. Subsequent works by Maino et al. [29] and Robert [37] further showed that SIDH can be broken in polynomial time even for arbitrary starting curves. These attacks rendered classical SIDH-based constructions, including SIKE, insecure.

At the same time, the attacks on SIDH revealed the power of higher dimensional techniques in isogeny-based cryptography. Kani's lemma allows one-dimensional isogenies to be embedded into higher-dimensional isogenies, and this viewpoint makes it possible, under suitable conditions, to recover an isogeny from its domain, codomain, degree, and auxiliary torsion information, without explicitly knowing its kernel generators. This perspective led to the higher-dimensional isogeny-based public-key encryption schemes, namely FESTA [7] and QFESTA [34]. Subsequently, higher-dimensional techniques were combined with SIDH-like structures to obtain more efficient schemes, including IS-CUBE [30], LIT-SiGamal [32], and POKÉ [6].

Among these schemes, POKÉ is currently one of the most efficient isogeny-based public-key encryption schemes. Its efficiency comes from the use of a hidden-degree isogeny: unlike IS-CUBE and LIT-SiGamal, whose security relies on long public-degree isogenies, POKÉ can work over smaller finite fields. This leads to compact public keys and ciphertexts and improves the overall performance. For example, at NIST security level I, BAT [18], one of the most compact lattice-based KEMs, has a ciphertext size of 473 bytes and a public-key size of 521 bytes, whereas POKÉ has a compressed ciphertext size of 312 bytes and a compressed public-key size of 270 bytes. This illustrates the strong compactness of POKÉ even compared with highly compact lattice-based KEMs.

However, the security foundation of POKÉ is less satisfactory than its efficiency. Its current security rationale is mainly based on the observation that all currently known key-recovery attacks appear to be ineffective. In particular, no reduction is known from key-recovery security to IND-CPA security. This is a familiar gap in SIDH-like constructions: as in the case of SIKE PKE, the hardness of recovering a party's secret isogeny does not by itself imply the computational hardness of recovering the shared secret.

Unlike SIDH-like schemes, group-action-based isogeny schemes provide a natural post-quantum analogue of Diffie–Hellman key exchange protocol. CSIDH [10] instantiates an efficient commutative class-group action for non-interactive key exchange, and CORAL [3] further explores this direction toward faster post-quantum NIKE. A feature of group-action-based schemes is that, in the Algebraic Group Action Model (AGAM) [19], shared-secret-recovery security can be reduced from key-recovery security, giving a cleaner security foundation than relying on ad-hoc computational assumptions. However, their commutative group structure also enables Kuperberg's subexponential-time quantum attack on the hidden shift problem [14], which makes CSIDH-like protocols use significantly larger parameters [10]. Consequently, despite their robust security, group-action-based schemes show slower performance than SIDH-like schemes.

Aardal, Basso, and Riepel [2] recently introduced the algebraic isogeny model (AIM) to address this type of gap. In the AIM, an *algebraic* algorithm is required

to provide an *explanation* isogeny

$$\phi : E_{\mathsf{known}} \to E_{\mathsf{output}}$$

whenever it outputs a curve E_{output}, where E_{known} is a curve that has already been provided to the algorithm. Although AIM was originally introduced to complete the quantum security proof of SQIsign [5], it also enables a reduction from key-recovery security to shared-secret-recovery security even in SIDH-like key exchange protocols.

The point-based structure of POKÉ, however, prevents this framework from applying directly. In POKÉ, the shared secret is not an elliptic curve but a pair of points obtained by evaluating torsion points through the relevant isogeny diagram. Since AIM controls how an adversary outputs new curves by requiring explanation isogenies, it does not naturally capture the security of a shared secret represented by points. Thus, a reduction from key-recovery security to its IND-CPA security is still infeasible in the AIM. LIT-SiGamal embeds messages into scalar multiplications of points, and therefore relies on point-based structures, making it unsuitable for applying the AIM framework. On the other hand, IS-CUBE shares secrets through curves and is therefore more compatible with AIM-based reasoning, but it requires primes of size roughly 8λ, making the scheme inefficient in practice.

1.1 Contributions

We propose a new isogeny-based PKE scheme called INKE, short for *INtermediate curve-based Key Exchange*. The main idea of INKE is to replace the point-based shared secret of POKÉ with an intermediate elliptic curve.

This modification addresses the limitation described above. Since the shared secret is now a curve, it can be handled through explanation isogenies in the AIM framework, enabling a reduction from IND-CPA security to key-recovery security. Our main contributions are as follows:

- **Construction of new practical isogeny-based PKE.** We propose a new practical isogeny-based PKE scheme, INKE. To the best of our knowledge, except for group-action-based protocols such as CSIDH [10] and CORAL [3], which admit reduction from key-recovery security to IND-CPA security in the algebraic group action model (AGAM), INKE is the first practical SIDH-like protocol whose IND-CPA security can be reduced solely to key-recovery security within an algebraic security model. Moreover, INKE is more efficient than existing group-action-based key exchange protocols. For example, CORAL requires a 2032-bit prime to achieve quantum security and takes approximately 240 ms per group-action evaluation, whereas INKE achieves encryption and decryption in approximately 54 ms and 62 ms in NIST level I, respectively.
- **Security proof of INKE in the Algebraic Isogeny Model.** We use the AIM to prove a quantum reduction from key-recovery security to IND-CPA

security. The framework of Aardal, Basso, and Riepel [2] provides a reduction from key-recovery security to shared-secret security. Their reduction applies when the starting curve is not fixed, and relies on the EndRing-to-OneEnd reduction of Page and Wesolowski [35].
However, when the starting curve is fixed to E_0, the EndRing-to-OneEnd reduction no longer applies, and the resulting shared-secret security relies on an additional non-standard assumption such as the hardness of the OneEnd problem on the parties' public curves. In this work, we address this issue by instead using the EndRing-to-Suborder quantum reduction of Mingjie Chen and Christophe Petit [13]. Consequently, we show that the IND-CPA security of INKE can be established solely from the key-recovery security assumptions in the AIM framework. The details of the reduction are given in Section 4.3.

– **Optimized C implementations and concrete comparison.** We provide optimized C implementations of both POKÉ and INKE, together with parameter sets for INKE targeting NIST security levels 1, 3, and 5. Implementing both schemes in the same environment and with the same level of optimization allows a fair comparison. Details are provided in Section 5, and our source code is available at

`https://github.com/gusgkr0117/poke`

This paper is organized as follows: In Section 2, we present the necessary background, including two-dimensional isogeny representations, the POKÉ algorithm, and its security. Section 3 introduces our proposed algorithm, followed by a security analysis in Section 4. Implementation results are provided in Section 5.

1.2 Technical Overview

In order to describe the main idea of INKE, we first review the structure of POKÉ. In POKÉ, key generation and decryption use a two-dimensional representation of an isogeny, while encryption uses a kernel representation of a fixed-degree isogeny. Let $E_0 : y^2 = x^3 + x$ be a supersingular elliptic curve defined over $\mathbb{F}_{p^2}$. Let $\phi_1 : E_0 \to E_{A_1}$ and $\phi_2 : E_{A_1} \to E_A$ be one-dimensional isogenies such that $\deg \phi_1 = q$ and $\deg \phi_2 = 2^a - q$ for some $q \in [0, 2^a)$, and suppose that they are embedded into the two-dimensional isogeny Φ of degree $(2^a, 2^a)$ defined by

$$\Phi = \begin{pmatrix} \phi_1 & \widehat{\phi}_2 \\ -\phi_3 & \widehat{\phi}_4 \end{pmatrix} : E_0 \times E_A \to E_{A_1} \times E_{A_2},$$

where the isogenies $\phi_3 : E_0 \to E_{A_2}$ and $\phi_4 : E_{A_2} \to E_A$ form an isogeny commutative diagram (also called an isogeny diamond) with ϕ_1 and ϕ_2, i.e.

$$\phi_2 = [\phi_1]_*\phi_3 \text{ and } \phi_4 = [\phi_3]_*\phi_1,$$

where $[\phi_1]_*\phi_3$, $[\phi_3]_*\phi_1$ denote the push-forward of ϕ_3 and the push-forward of ϕ_1 by ϕ_1 and ϕ_3, respectively, and $\widehat{\phi}_2$ and $\widehat{\phi}_4$ denote the duals of ϕ_2 and ϕ_4,

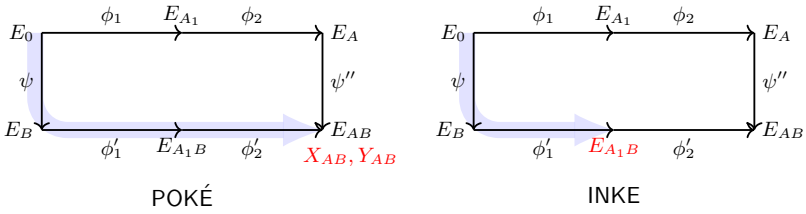

Fig. 1: Comparison of the shared-secret derivation in POKÉ and INKE. The shared secret is colored by red.

respectively. Let (P_0, Q_0) be a 2^a-torsion basis of E_0 and (R_0, S_0) be a 3^b-torsion basis of E_0. The public key is the curves E_0, E_A together with masked 2^a-torsion basis $P_A, Q_A \in E_A$ and masked 3^b-torsion basis $R_A, S_A \in E_0$. For encryption, the 3^b-degree isogenies $\psi : E_0 \to E_B$ and $\psi'' : E_A \to E_{AB}$ are chosen uniformly at random. The overall structure is illustrated in the left diagram of Figure 1.

The ciphertext is computed by xoring the message with the hash of a pair of evaluated torsion points (X_{AB}, Y_{AB}) obtained through a commutative isogeny diagram (see the highlighted thick path in the left diagram of Figure 1), i.e.

$$(X_{AB}, Y_{AB})^T = D_5 \cdot (\psi'' \circ \phi_2 \circ \phi_1(X_0), \psi'' \circ \phi_2 \circ \phi_1(Y_0))^T,$$
$$\mathsf{ct} = \mathsf{msg} \oplus H(X_{AB} || Y_{AB}),$$

where $D_5 \in M_{2\times 2}(\mathbb{Z}/5^c\mathbb{Z})$, and (X_0, Y_0) is a 5^c-torsion basis of E_0.

The main idea behind INKE is to avoid using torsion points for making a ciphertext. Instead of using points, INKE uses the intermediate elliptic curve E_{A_1B} in the encryption. This curve is obtained by computing isogenies according to the highlighted path in the right diagram of Figure 1.

Using elliptic curves rather than torsion points in encryption provides additional capability to its IND-CPA adversary in the algebraic isogeny model, since the adversary can produce an isogeny path to the curve it outputs. This enables a quantum reduction from key-recovery security to IND-CPA security within the AIM framework, yielding a more robust security foundation than point-based constructions such as POKÉ.

1.3 Related Works

Algebraic Isogeny Model Recently, Aardal, Basso, and Riepel introduced the algebraic isogeny model (AIM), a general security framework for isogeny-based cryptography in [2]. The AIM generalizes the algebraic group action model (AGAM), and every result established in the AGAM can be lifted to the AIM, showing that AIM is strictly more general.

The security of SQIsign has been proved relying on the hint-based endomorphism ring problem introduced in [1]. However, its quantum security was not previously established, since classical rewinding techniques require a unique-response property in order to extract the secret key from a quantum adversary. In

the AIM framework, rewinding is no longer necessary, as an algebraic adversary is required to provide an explanation isogeny associated with every output elliptic curve. This property enables the quantum security proof of SQIsign within the AIM framework.

The authors also show that a reduction from key-recovery security to shared-secret security can be achieved in their AIM framework. Their reduction applies when the starting curve is not fixed, and relies on the EndRing-to-OneEnd reduction of Page and Wesolowski [35]. However, as noted in [35], when the starting curve is fixed to E_0, the EndRing-to-OneEnd reduction is not possible.

PIKE: another variant of POKÉ Recently, Cai, Chen, Lai, and Lin proposed PIKE [8] (PKC 2026), a variant of POKÉ. PIKE was developed independently and follows a fundamentally different approach from INKE. PIKE avoids discrete logarithm computations by replacing torsion points with pairing values for making a ciphertext. This design improves efficiency while also maintaining the sizes of the public keys and ciphertexts small. In Section 5, we provide a new detailed comparison[4] based on our latest optimized implementation.

Since PIKE uses pairing values in encryption, its security does not naturally fit within the AIM framework, and no reduction from key-recovery security to its IND-CPA security is currently known.

Recent attack on INKE Recently, Moriya [31] showed that there exist weak keys in POKÉ-like schemes, reducing the complexity of certain attacks. Although POKÉ-2D and PIKE remains secure against this attack, both POKÉ-4D and INKE are affected, requiring larger parameter sizes to maintain the targeted security level. In this work, we regard this attack as effective and accordingly revise our security parameters. We also update our implementation and performance analysis in Section 5 using the revised parameters, and compare the resulting performance with PIKE and POKÉ.

2 Preliminaries

This section provides the background necessary for the paper. First, a two-dimensional representation of an isogeny is presented. Then, POKÉ algorithm is introduced, which is the primary focus of our optimization in this paper.

Throughout this work, p denotes a prime number, and we consider supersingular elliptic curves defined over $\mathbb{F}_{p^2}$. For a point P on an elliptic curve E, the smallest positive integer ℓ such that $[\ell]P = \mathcal{O}$ is called the order of P. The subgroup

$$E[\ell] := \{P \in E : [\ell]P = \mathcal{O}\}$$

[4] The initial version of INKE was publicly released on ePrint prior to PIKE, and the PIKE paper included performance comparisons between PIKE, POKÉ, and INKE based on a modified version of our C implementation released with the earlier ePrint version. Since then, however, we have incorporated several additional optimizations into the implementation of INKE and POKÉ.

is called the ℓ-torsion subgroup of E.

An isogeny $\phi : E_1 \to E_2$ is a non-constant morphism between elliptic curves that is also a group homomorphism between the elliptic curve groups. When $E_1 \cong E_2$, the map ϕ is called an endomorphism of E_1, and we write $\phi \in \mathrm{End}(E_1)$. The multiplication-by-$m$ map is a trivial endomorphism, denoted by $[m]$.

The degree of an isogeny is its degree as a rational map, which coincides with the size of its kernel when the isogeny is separable. For every isogeny $\phi : E_1 \to E_2$, there exists an isogeny $\widehat{\phi} : E_2 \to E_1$ such that

$$\widehat{\phi} \circ \phi = [\deg \phi].$$

The map $\widehat{\phi}$ is called the dual isogeny of ϕ. We refer the reader to [39] for further details on elliptic curves and isogenies.

Let $\phi_1 : E_0 \to E_1$ and $\phi_2 : E_0 \to E_2$ be two isogenies with the same domain curve E_0 and coprime degrees. We denote $[\phi_1]_*\phi_2 : E_1 \to E_{12}$ as a pushforward of ϕ_2 under ϕ_1, i.e. $\ker([\phi_1]_*\phi_2) = \phi_1(\ker \phi_2)$. Then, we have two sets of parallel isogenies $\phi_1 : E_0 \to E_1$, $\phi_2 : E_0 \to E_2$, $[\phi_1]_*\phi_2 : E_1 \to E_{12}$ and $[\phi_2]_*\phi_1 : E_2 \to E_{12}$ which form a commutative diagram.

2.1 Two-Dimensional Representation

Theorem 1 (Kani's Lemma [25]). *Consider the following commutative diagram of isogenies*

$$\begin{array}{ccc} E_0 & \xrightarrow{\phi_1} & E_1 \\ {\scriptstyle \phi_2}\downarrow & & \downarrow{\scriptstyle \phi_2'} \\ E_2 & \xrightarrow[\phi_1']{} & E_{12} \end{array}$$

where $\deg(\phi_i) = \deg(\phi_i') = d_i$ *and* $\gcd(d_1, d_2) = 1$. *Then, the isogeny*

$$\Phi = \begin{pmatrix} \phi_1 & -\hat{\phi}_2' \\ \phi_2 & \hat{\phi}_1' \end{pmatrix} : E_0 \times E_{12} \to E_1 \times E_2$$

is a $(d_1 + d_2)$*-isogeny whose kernel is given by*

$$\mathrm{Ker}\Phi = \{([-d_1]P, \phi_2' \circ \phi_1(P)) | P \in E_0[d_1 + d_2]\}.$$

Kani's lemma enables the following two-dimensional representation of a certain type of isogenies of degree $q(2^a - q)$ used in FESTA [7] and QFESTA [34]. Let E be an elliptic curve such that $2^a | \#E(\mathbb{F}_{p^2})$. An isogeny $\phi : E \to E'$ admits an efficient representation

$$(q, E, E', (P_0, Q_0), (\phi(P_0), \phi(Q_0)))$$

where $\deg(\phi) = q(2^a - q)$, $q < 2^a$, $\gcd(q, 2) = 1$ and a basis $\langle P_0, Q_0 \rangle = E[2^a]$. This is called a *two-dimensional representation of an isogeny* ϕ.

Algorithm 1 describes how we can evaluate a point $R \in E$ using a two-dimensional representation. There exists an efficient algorithm ([15]) to compute a two-dimensional isogeny Φ when $\deg \Phi$ is a power of two. For the codomain computation and point evaluations of the two-dimensional isogeny in Algorithm 1, we refer to [15, Algorithm 5 and Algorithm 6].

Algorithm 1: Evaluation of a two-dimensional representation

Input: A two-dimensional representation $(q, E, E', (P_0, Q_0), (\phi(P_0), \phi(Q_0))$ and a point $R \in E(\mathbb{F}_{p^2})$ such that $\gcd(\mathrm{ord}(R), 2) = 1$
Output: $\phi(R)$

1 Compute Φ with kernel $\langle([-q]P_0, \phi(P)), ([-q]Q_0, \phi(Q_0))\rangle$.
2 Evaluate $(R', _) = \Phi(R, \mathcal{O}_E)$.
3 Let C be the order of R.
4 Generate a basis U, V of $E'[C]$.
5 Evaluate $(U', _) = \Phi(\mathcal{O}_{E'}, U)$.
6 Evaluate $(V', _) = \Phi(\mathcal{O}_{E'}, V)$.
7 Find x, y such that $R' = [x]U' + [y]V'$. `// By solving ECDLP`
8 Compute $\phi(R) = [2^a - q]([x]U + [y]V)$.
9 **return** $\phi(R)$.

In line 7 of Algorithm 1, evaluating a point under a two-dimensional representation requires solving a discrete logarithm, which becomes costly when the point has a large non-smooth order.

2.2 POKÉ Protocol

In this section, we briefly introduce the POKÉ protocol. It is based on the SIDH-like structure, which takes place between two parties, Alice and Bob. Alice can compute an isogeny from its domain and codomain curves by using Kani's theorem, while Bob only computes an isogeny represented by a single curve. To avoid the torsion point attack, every point evaluation must involve masking procedures.

Let p be a prime of the form $p = 2^a 3^b 5^c f - 1$ where f is a small cofactor. Let E_0 be the supersingular curve defined over $\mathbb{F}_p$ with j-invariant 1728. Let P_0, Q_0 be a canonical basis of $E_0[2^a]$, and R_0, S_0 be a canonical basis of $E_0[3^b]$. Let X_0, Y_0 be a canonical basis of $E_0[5^c]$. Let KDF be a key derivation function.

Key generation The receiver (Alice) samples a random value $q \in [0, 2^a)$ coprime with 30, and computes a random isogeny $\phi : E_0 \to E_A$ of degree $q(2^a - q)$ by using Algorithm 2. In POKÉ, a rigorous algorithm [6, Algorithm 2] is also proposed to sample $q(2^a - q)$-isogenies with a distribution identical to the uniform random $q(2^a - q)$-isogeny distribution. We employ the heuristic version for the efficiency which we call Gen22Isog (Algorithm 2). As the distribution of

Gen22Isog depends on the algorithm $\mathsf{FullRepresentInteger}_{\mathcal{O}_0}$ ([16, Algorithm 1]), it is assumed that the output of $\mathsf{FullRepresentInteger}_{\mathcal{O}_0}$ is well distributed in $\mathcal{O}_0$.

Algorithm 2: $\mathsf{Gen22Isog}(p, q)$ (heuristic)

Input: A degree q, a prime p of the form $p = 2^a 3^b f - 1$.
Output: A representation of pair of isogenies (ϕ_1, ϕ_2) such that $\phi_2 \circ \phi_1 : E_0 \to E_A$ of degree $q(2^a - q)$.

Generate an endomorphism $\theta \leftarrow \mathsf{FullRepresentInteger}_{\mathcal{O}_0}(q(2^a - q) \cdot 3^b)$.
// ▷ Using [16, Algorithm 1]
Compute $P', Q' = \theta(P_0), \theta(Q_0)$.
Let $K = \ker(\hat{\theta}) \cap E_0[3^b]$.
Compute $\eta : E_0 \to E_A$ with kernel K.
Compute $P_A, Q_A = [1/(3^b)]\eta(P'), [1/(3^b)]\eta(Q')$.
Compute Φ with kernel $\langle([-q]P_0, P_A), ([-q]Q_0, Q_A)\rangle$.
return (ϕ_1, ϕ_2) where $\Phi = \begin{pmatrix} \phi_1 & \widehat{\phi}_2 \\ * & * \end{pmatrix}$, $\deg \phi_1 = q$ and $\deg \phi_2 = 2^a - q$.

Algorithm 3: Key Generation

Input: $E_0, P_0, Q_0 \in E_0[2^a], R_0, S_0 \in E_0[3^b], X_0, Y_0 \in E_0[5^c]$.
Output: A secret/public key pair.

Sample a random $q \in [0, 2^a]$ such that $\gcd(q(2^a - q), 30) = 1$.
Generate a $q(2^a - q)$-isogeny $\phi = \phi_2 \circ \phi_1 : E_0 \to E_A$ from $(\phi_1, \phi_2) \leftarrow \mathsf{Gen22Isog}(p, q)$. // Algorithm 2
Sample random $\alpha_2, \beta_2 \xleftarrow{\$} \mathbb{Z}_{2^a}^{\times}$.
Sample a random $\gamma_3 \leftarrow \mathbb{Z}_{3^b}^{\times}$ and a random $\delta_5 \leftarrow \mathbb{Z}_{5^c}^{\times}$.
Compute $P_A, Q_A = [\alpha_2]\phi(P_0), [\beta_2]\phi(Q_0)$.
Compute $R_A, S_A = [\gamma_3]\phi(S_0), [\gamma_3]\phi(R_0)$.
Compute $X_A, Y_A = [\delta_5]\phi(X_0), [\delta_5]\phi(Y_0)$.
return $\mathsf{sk} = (q, \alpha_2, \beta_2, \delta_5)$, $\mathsf{pk} = (E_A, P_A, Q_A, R_A, S_A, X_A, Y_A)$.

Encryption The sender (Bob) samples a random integer $r_3 \in \mathbb{Z}_{3^b}$ and computes two isogenies ψ and ψ' of the kernel $\langle R_0 + [r_3]S_0\rangle$ and $\langle R_A + [r_3]S_A\rangle$, respectively. After generating a shared secret X_{AB}, Y_{AB} from a commutative diagram, it makes the ciphertext $\mathsf{ct} := m \oplus \mathsf{KDF}(X_{AB}, Y_{AB})$ for some fixed key-derivation function KDF. More precisely, encryption is performed as Algorithm 4.

Algorithm 4: Encryption

Input: A message m, a public key $E_A, P_A, Q_A, R_A, S_A, X_A, Y_A$.
Output: A ciphertext $\mathtt{ct}$.

Sample a random $r_3 \leftarrow \mathbb{Z}_{3^b}$.
Compute the isogeny $\psi : E_0 \to E_B$ with kernel $\langle R_0 + [r_3]S_0 \rangle$.
Compute the isogeny $\psi' : E_A \to E_{AB}$ with kernel $\langle R_A + [r_3]S_A \rangle$.
Sample a random $\omega_2 \leftarrow \mathbb{Z}_{2^a}^{\times}$.
Compute $P_B = [\omega_2]\psi(P_0)$, $Q_B = [1/\omega_2]\psi(Q_0)$.
Compute $P_{AB} = [\omega_2]\psi'(P_A)$, $Q_{AB} = [1/\omega_2]\psi'(Q_A)$.
Sample a random $D_5 \leftarrow \mathrm{SL}_2(\mathbb{Z}_{5^c})$.
Compute $[X_B, Y_B]^T = D_5 \begin{bmatrix} \psi(X_0) \\ \psi(Y_0) \end{bmatrix}$.
Compute $[X_{AB}, Y_{AB}]^T = D_5 \begin{bmatrix} \psi'(X_A) \\ \psi'(Y_A) \end{bmatrix}$.
Compute $\mathsf{ct}' = \mathsf{KDF}(X_{AB}, Y_{AB}) \oplus m$.
return $\mathsf{ct} = ((E_B, P_B, Q_B, X_B, Y_B), (E_{AB}, P_{AB}, Q_{AB}), \mathsf{ct}')$.

Decryption The receiver computes the points P'_{AB} and Q'_{AB} from the masking value α_2 and β_2. Then, it recovers the shared secret X_{AB}, Y_{AB} by computing the two-dimensional isogeny Φ' and extracting the message as $m = \mathsf{ct} \oplus \mathsf{KDF}(X_{AB}, Y_{AB})$.

Algorithm 5: Decryption

Input: $\mathsf{sk} = (q, \alpha_2, \beta_2, \delta_5)$,
$\mathtt{ct} = ((E_B, P_B, Q_B, X_B, Y_B), (E_{AB}, P_{AB}, Q_{AB}), \mathsf{ct}')$.
Output: A message m'.

Compute $P'_{AB} = [1/\alpha_2]P_{AB}, \quad Q'_{AB} = [1/\beta_2]Q_{AB}$.
Compute Φ' with kernel $\langle ([-q]P_B, P'_{AB}), ([-q]Q_B, Q'_{AB}) \rangle$.
Evaluate $(X'_B, _) = \Phi'(X_B, \mathcal{O}_{E_{AB}})$.
Evaluate $(Y'_B, _) = \Phi'(Y_B, \mathcal{O}_{E_{AB}})$.
Generate a basis U, V of $E_{AB}[5^c]$.
Evaluate $(U', _) = \Phi'(\mathcal{O}_{E_B}, U)$.
Evaluate $(V', _) = \Phi'(\mathcal{O}_{E_B}, V)$.
Find x, y such that $X'_B = [x]U' + [y]V'$.
Find w, z such that $Y'_B = [w]U' + [z]V'$.
Compute $X_{AB} = [\delta_5][2^a - q]([x]U + [y]V)$.
Compute $Y_{AB} = [\delta_5][2^a - q]([w]U + [z]V)$.
Compute $m' = \mathsf{KDF}(X_{AB}, Y_{AB}) \oplus \mathsf{ct}'$.
return m'.

2.3 POKÉ Security

POKÉ has been shown to be IND-CPA secure under the hardness assumption of the custom problem called C-POKE (Problem 1).

Problem 1 (C-POKE). Let E_0 be a supersingular elliptic curve defined over $\mathbb{F}_{p^2}$, with known endomorphism ring. Let P_0, Q_0 be a basis of $E_0[2^a]$, R_0, S_0 a basis of $E_0[3^b]$, and X_0, Y_0 a basis of $E_0[5^c]$.

Let $\phi : E_0 \to E_A$ be an isogeny of degree $q(2^a - q)$, for some unknown value $q \in [0, 2^a]$. Write

$$\begin{aligned} P_A, Q_A &= [\alpha_2]\,\phi(P_0), \quad [\beta_2]\,\phi(Q_0), \\ R_A, S_A &= [\gamma_3]\,\phi(R_0), \quad [\gamma_3]\,\phi(S_0), \\ X_A, Y_A &= [\delta_5]\,\phi(X_0), \quad [\delta_5]\,\phi(Y_0), \end{aligned}$$

where $\alpha_2, \beta_2, \gamma_3, \delta_5$ are uniformly random scalars from

$$\mathbb{Z}_{2^a}^{\times} \times \mathbb{Z}_{2^a}^{\times} \times \mathbb{Z}_{3^b}^{\times} \times \mathbb{Z}_{5^c}^{\times}.$$

Let $\psi : E_0 \to E_B$ be an isogeny of degree 3^b, and write $\psi' : E_A \to E_{AB}$ for its pushforward $\phi_*\psi$. Write

$$\begin{aligned} P_B, Q_B &= [\omega_2]\,\psi(P_0), \quad [1/\omega_2]\,\psi(Q_0), \\ P_{AB}, Q_{AB} &= [\omega_2]\,\psi'(P_A), \quad [1/\omega_2]\,\psi'(Q_A), \\ X_B, Y_B &= \mathbf{D}_5 \left[\psi(X_0), \psi(Y_0)\right]^T, \\ X_{AB}, Y_{AB} &= \mathbf{D}_5 \left[\psi'(X_A), \psi'(Y_A)\right]^T, \end{aligned}$$

where ω_2 is a uniformly random scalar from $\mathbb{Z}_{2^a}^{\times}$, and $\mathbf{D}_5$ is a uniformly random matrix from $\mathrm{SL}_2(\mathbb{Z}_{5^c})$.

Given $(E_0, P_0, Q_0), (R_0, S_0), (X_0, Y_0)$, $(E_A, P_A, Q_A), (R_A, S_A), (X_A, Y_A)$, $(E_B, P_B, Q_B), (X_B, Y_B)$, and (E_{AB}, P_{AB}, Q_{AB}), compute the points X_{AB}, Y_{AB}.

Although POKÉ is IND-CPA secure assuming the computational hardness of C-POKE, in [6], the hardness of the C-POKE problem is not analyzed directly, leaving the security of POKÉ unclear. Instead, the authors argue that POKÉ is secure against currently known and predictable key-recovery attacks, which provided some evidence of the plausibility of the design. However, no reduction from key-recovery security to the hardness of the C-POKE problem is provided.

3 INKE PKE protocol

Our new scheme resembles the structure of POKÉ, except that we replace the 5^c-torsion points in POKÉ with the intermediate curve of the two-dimensional representation of the secret isogeny ϕ.

In our construction, the receiver (Alice) reveals the intermediate curve and the evaluated 3^b-torsion basis on that curve. Since the torsion basis is masked and the secret degree q remains hidden, revealing the intermediate curve E_{A_1}, and its associated torsion information does not compromise security as long as the degree q is sufficiently large. Conceptually, our scheme can be viewed as exposing an additional isogeny and its codomain curve, rather than revealing additional torsion points such as (X_A, Y_A) as done in POKÉ. A detailed description of our protocol is provided below. The overall process is illustrated in Figure 2.

Key generation Given a random value $q \in [0, 2^a)$ such that $q \equiv 5 \pmod 6$, we first sample a random isogeny $\phi : E_0 \to E_A$ of degree $q(2^a - q)$. The authors of POKÉ propose two approaches for sampling such an isogeny: a rigorous approach and a heuristic one. In our case, we adopt the heuristic approach represented in Algorithm 2 for efficiency. The two-dimensional representation of the sampled isogeny $\phi : E_0 \to E_A$ corresponds to an isogeny commutative diagram involving E_0 and E_A, along with two intermediate curves E_{A_1} and E_{A_2} such that $\Phi : E_0 \times E_A \to E_{A_1} \times E_{A_2}$ for the corresponding two-dimensional isogeny Φ.

In our construction, we leverage one of these intermediate curves E_{A_1} or E_{A_2} as a public key of a SIDH-like key exchange protocol. Suppose that E_{A_1} is used. The curve E_A is used as an auxiliary curve to compute the isogeny $\phi_1 : E_0 \to E_{A_1}$. Consequently, our key generation procedure requires a larger public key compared to that of POKÉ, as it additionally includes the intermediate curve E_{A_1} and a 3^b-torsion basis (R_{A_1}, S_{A_1}) on this curve. To mitigate the torsion point attacks, the bases (R_{A_1}, S_{A_1}) and (R_A, S_A) are masked independently using random scalars γ_3' and γ_3, respectively. Since our scheme employs a j-invariant of an elliptic curve as the shared secret rather than points, our key generation algorithm avoids the use of 5^c-torsion points, which saves some computational overhead.

Algorithm 6: Key Generation

Input: E_0, $P_0, Q_0 \in E_0[2^a]$, $R_0, S_0 \in E_0[3^b]$
Output: A secret/public key pair

Sample a random $q \in [0, 2^a)$ such that $q \equiv 5 \pmod 6$.
Sample a pair of isogenies $(\phi_1, \phi_2) \leftarrow \mathsf{Gen22Isog}(p, q)$ and write $\phi = \phi_2 \circ \phi_1$.
// Algorithm 2
Sample random $\alpha_2, \beta_2 \leftarrow \mathbb{Z}_{2^a}^{\times}$.
Sample a random $\gamma_3, \gamma_3' \leftarrow \mathbb{Z}_{3^b}^{\times}$.
Compute $P_A, Q_A = [\alpha_2]\phi(P_0),\ [\beta_2]\phi(Q_0)$.
Compute $R_A, S_A = [\gamma_3]\phi(S_0),\ [\gamma_3]\phi(R_0)$.
Compute E_{A_1} and $R_{A_1}, S_{A_1} = [\gamma_3']\phi_1(R_0),\ [\gamma_3']\phi_1(S_0)$.
return $\mathsf{sk} = (q, \alpha_2, \beta_2)$, $\mathsf{pk} = (E_A, P_A, Q_A, R_A, S_A, E_{A_1}, R_{A_1}, S_{A_1})$.

Encryption In our encryption procedure, we compute three 3^b-isogenies: ψ, ψ' and ψ''. Compared to POKÉ, our encryption performs an additional isogeny to compute the codomain curve E_{A_1B}, starting from the intermediate curve E_{A_1}, which lies parallel to ψ and ψ'' in the isogeny commutative diagram. Since we use the j-invariant of E_{A_1B} as the shared secret, no additional point evaluation is needed.

Algorithm 7: Encryption

Input: A message m, a public key $E_A, P_A, Q_A, R_A, S_A, E_{A_1}, R_{A_1}, S_{A_1}$.
Output: A ciphertext ct.

Sample a random $r_3 \leftarrow \mathbb{Z}_{3^b}$.
Compute the isogeny $\psi : E_0 \to E_B$ with kernel $\langle R_0 + [r_3]S_0 \rangle$.
Compute the isogeny $\psi' : E_{A_1} \to E_{A_1B}$ with kernel $\langle R_{A_1} + [r_3]S_{A_1} \rangle$.
Compute the isogeny $\psi'' : E_A \to E_{AB}$ with kernel $\langle R_A + [r_3]S_A \rangle$.
Sample a random $\omega_2 \leftarrow \mathbb{Z}_{2^a}^{\times}$.
Compute $P_B = [\omega_2]\psi(P_0)$, $Q_B = [1/\omega_2]\psi(Q_0)$.
Compute $P_{AB} = [\omega_2]\psi''(P_A)$, $Q_{AB} = [1/\omega_2]\psi''(Q_A)$.
Compute $\mathsf{ct}' = \mathsf{KDF}(j(E_{A_1B})) \oplus m$.
return $\mathsf{ct} = ((E_B, P_B, Q_B), (E_{AB}, P_{AB}, Q_{AB}), \mathsf{ct}')$.

Decryption In the decryption procedure, given a ciphertext

$$((E_B, P_B, Q_B), (E_{AB}, P_{AB}, Q_{AB}, \mathsf{ct}')),$$

we first recover the points $(P'_{AB}, Q'_{AB}) = ([1/\alpha_2]P_{AB}, [1/\beta_2]Q_{AB})$. To obtain the shared secret curve E_{A_1B}, we compute the two-dimensional isogeny $\Phi' : E_B \times E_{AB} \to E_{A_1B} \times F$ with the kernel

$$\langle ([-q]P_B, P'_{AB}), ([-q]Q_B, Q'_{AB}) \rangle,$$

and recover the message m' from ct' with $j(E_{A_1B})$.

Algorithm 8: Decryption

Input: $\mathsf{sk} = (q, \alpha_2, \beta_2)$, $\mathsf{ct} = ((E_B, P_B, Q_B), (E_{AB}, P_{AB}, Q_{AB}), \mathsf{ct}')$
Output: A message m'

Compute $P'_{AB} = [1/\alpha_2]P_{AB}, \quad Q'_{AB} = [1/\beta_2]Q_{AB}$
Compute Φ' of the kernel $\langle ([-q]P_B, P'_{AB}), ([-q]Q_B, Q'_{AB}) \rangle$
if $\mathrm{Im}(\Phi')$ *is not splitting* **then abort**
Compute the codomain curve $E_{A_1B} \times F$ of Φ'.
Compute $m' = \mathsf{KDF}(j(E_{A_1B})) \oplus \mathsf{ct}'$
return m'.

The entire keygen, encryption, and decyption procedures are depicted in Figure 2. Our diagram is very similar to the diagram of POKÉ [6, Figure 2], except that we use the intermediate curves E_{A_1} and E_{A_1B}.

How to identify E_{A_1B}? During the decryption process, we compute a $(2^a, 2^a)$-isogeny $\Phi' : E_A \times E_{AB} \to E_{A_1B} \times F_B$ and we must identify the right curve E_{A_1B} which corresponds to the curve E_{A_1}, i.e. $j(\mathrm{Im}(\phi_{1*}\psi)) = j(E_{A_1B})$ in Figure 2. A straightforward approach is to evaluate two points and compare the pairings.

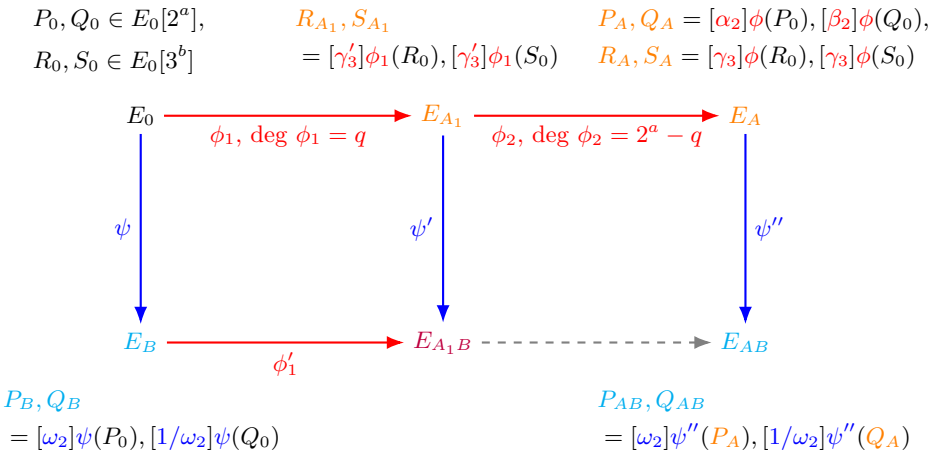

Fig. 2: Our INKE protocol. Public parameters are in black; elements in red are part of the secret key or computed during decryption; orange parts are public keys; blue are ephemeral secrets; light blue are ciphertext parts; purple shows the shared secret.

For example, we choose a random N-torsion basis $U_1, V_1 \in E_B$ and evaluate these two points by $(U_2, U_3) = \Phi'(U_1, \mathcal{O})$ and $(V_2, V_3) = \Phi'(V_1, \mathcal{O})$. Then, we can distinguish the two curves E_{A_1B} and F_B by solving the discrete logarithm $e(U_2, V_2)^x = e(U_1, V_1)$ and $e(U_3, V_3)^y = e(U_1, V_1)$ and obtaining $x \equiv \deg\phi_1 \mod N$ and $y \equiv \deg\phi_2 \mod N$. We choose the smallest N such that $\deg\phi_1 \pmod N$ and $\deg\phi_2 \pmod N$ are different. This method requires pairing and discrete logarithm computations, which are quite expensive.

Fortunately, the most efficient $(2^a, 2^a)$-isogeny computation based on the theta model [15] allows us to naturally distinguish the two curves without additional pairing computations. In the theta-model-based approach, isogeny computation relies on extra 4-torsion points to construct a theta structure and to work with the associated theta (Heisenberg) group. The automorphisms preserving a fixed theta structure induce symplectic automorphisms on the associated theta group [15]. The action of the symplectic automorphism on theta coordinates is constrained and does not allow switching of the two curves in an elliptic curve product. As a consequence, an intrinsic ordering is imposed on an elliptic curve product in the theta model. When traversing a two-dimensional isogeny diamond and returning to the starting surface, the induced transformation is a symplectic automorphism, which preserves this ordering of the elliptic curve product. From this perspective, the theta model can be viewed as providing a clean alternative to explicit pairing computations with extra torsion points, encoding the relevant information directly through the structure of the theta group.

4 Security of Our Scheme

In this section, we analyze the security of our scheme, which relies on the hardness of the computational problem that asks to find the shared secret given all public details. In our setting, the underlying hardness assumption is formalized as Definition 1, C-INKE, which closely resembles the C-POKE problem introduced in [6], except that the 5^c-torsion information is omitted and additional 3^b-torsion information, along with a new intermediate curve E_{A_1}, is given.

Definition 1 (C-INKE). *For an algorithm $\mathcal{A}$, we define the advantage of finding the codomain curve of the isogeny $\phi_{1*}\psi$, given the output of Experiment 1 as follows:*

$$\mathsf{Adv}^{\mathsf{C\text{-}INKE}} := \Pr[j(\mathrm{Im}(\phi_{1*}\psi)) = j' : j' \leftarrow \mathcal{A}(x) \wedge x \leftarrow \mathsf{C\text{-}INKE}(1^\lambda)]$$

Experiment 1: C-INKE Problem

$\underline{\mathsf{C\text{-}INKE}(1^\lambda)}$

1. Let $q \xleftarrow{\$} [0, 2^a)$ such that $q \equiv 5 \pmod 6$.
2. Sample a pair of isogenies $(\phi_1, \phi_2) \leftarrow \mathsf{Gen22Isog}(p, q)$ and let $\phi = \phi_2 \circ \phi_1 : E_0 \to E_A$.
3. Choose uniformly random scalars $(\alpha_2, \beta_2, \gamma_3, \gamma_3') \in \mathbb{Z}_{2^a}^\times \times \mathbb{Z}_{2^a}^\times \times \mathbb{Z}_{3^b}^\times \times \mathbb{Z}_{3^b}^\times$.
4. Compute masked bases $(P_A, Q_A) = ([\alpha_2]\phi(P_0), [\beta_2]\phi(P_0))$, $(R_A, S_A) = ([\gamma_3]\phi(R_0), [\gamma_3]\phi(S_0))$ and $(R_{A_1}, S_{A_1}) = ([\gamma_3']\phi_1(R_0), [\gamma_3']\phi_1(S_0))$.
5. Choose a uniformly random cyclic isogeny $\psi : E_0 \to E_B$ of degree 3^b.
6. Let $\psi'' = \phi_*\psi : E_A \to E_{AB}$.
7. Choose a uniformly random scalar $\omega \in [0, 2^a)$.
8. Compute masked bases $(P_B, Q_B) = ([\omega_2]\psi(P_0), [1/\omega_2]\psi(Q_0))$ and $(P_{AB}, Q_{AB}) = ([\omega_2]\psi''(P_A), [1/\omega_2]\psi''(Q_A))$.
9. Return $(E_0, (P_0, Q_0), (R_0, S_0), E_{A_1}, (R_{A_1}, S_{A_1}), E_A, (P_A, Q_A), (R_A, S_A), E_B, (P_B, Q_B), E_{AB}, (P_{AB}, Q_{AB})$

Theorem 2. *Suppose that the key-derivation function of INKE is modeled as a random oracle. For any PPT adversary $\mathcal{A}$ for the IND-CPA of INKE in a random oracle model, there exists a PPT algorithm $\mathcal{B}$ such that*

$$\mathsf{Adv}^{\mathsf{IND\text{-}CPA}}_{\mathsf{INKE}}(\mathcal{A}) \leq q_g \cdot \mathsf{Adv}^{\mathsf{C\text{-}INKE}}(\mathcal{B}),$$

where q_g is the number of random oracle queries.

The proof can be found in Appendix C.1.

It is also possible to construct an IND-CCA secure KEM in the Quantum Random Oracle Model (QROM) by applying the Fujisaki-Okamoto (FO) transformation [20], combined with the modular proof framework of [22, Theorem 3.2 and Theorem 3.3].

4.1 Analysis of Receiver's Security

The following problem is our modified version of the receiver's security of POKÉ [6, Problem 11] for security against key-recovery attacks.

Definition 2 (Receiver's Security). *For an algorithm $\mathcal{A}$, we define the advantage of finding an efficient representation of ϕ, given the output of* $\mathsf{Recv}(1^\lambda)$ *(Experiment 2) as* $\mathsf{Adv}^{\mathsf{Recv}}(\mathcal{A})$.

Experiment 2: Receiver's Security

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

1. Choose a random integer $q \in [0, 2^a)$ such that $q \equiv 5 \pmod 6$.
2. Sample a pair of isogenies $(\phi_1, \phi_2) \leftarrow \mathsf{Gen22Isog}(p, q)$ and let $\phi = \phi_2 \circ \phi_1 : E_0 \to E_A$.
3. Choose uniformly random scalars $(\alpha_2, \beta_2, \gamma_3, \gamma_3') \in \mathbb{Z}_{2^a}^\times \times \mathbb{Z}_{2^a}^\times \times \mathbb{Z}_{3^b}^\times \times \mathbb{Z}_{3^b}^\times$.
4. Compute masked bases

$$\begin{aligned}(P_A, Q_A) &= ([\alpha_2]\phi(P_0), [\beta_2]\phi(Q_0))\\ (R_A, S_A) &= ([\gamma_3]\phi(R_0), [\gamma_3]\phi(S_0))\\ (R_{A_1}, S_{A_1}) &= ([\gamma_3']\phi_1(R_0), [\gamma_3']\phi_1(S_0)).\end{aligned}$$

5. Return $E_0, (P_0, Q_0), (R_0, S_0), E_{A_1}, (R_{A_1}, S_{A_1}), E_A, (P_A, Q_A), (R_A, S_A)$.

The receiver's security of INKE can be viewed as a variant of that of POKÉ, with additional information of an intermediate curve E_{A_1}, and its basis (R_1, S_1). Consequently, all aspects of the receiver's security analysis for POKÉ in [6, Section 6.1] apply directly to INKE as well. For example, in POKÉ it is shown that without diagonal scaling, masking $\phi(P_0)$ and $\phi(Q_0)$ leaks information about the isogeny, while with diagonal scaling the isogeny is perfectly concealed. This observation equally applies in the INKE setting. In this section, we focus only on the additional issues that may arise in INKE due to the exposure of the intermediate curve.

Exploiting the two-dimensional torsion information Since the degree of ϕ_1 is not fixed, any attack using the torsion information associated with ϕ_1 is infeasible. Therefore, it is natural to consider exploiting torsion information in the higher-dimensional setting, since the two-dimensional isogeny Φ has a fixed degree of $(2^a, 2^a)$.

From the perspective of the two-dimensional isogeny $\Phi : E_0 \times E_2 \to E_1 \times F$, the setting of Problem 2 can be interpreted as a partial exposure of the masked torsion information $[\gamma_3']\Phi((R_0, 0_{E_2}))$ and $[\gamma_3']\Phi((S_0, 0_{E_2}))$, with the degree of Φ known to be 2^a.

Given the abelian surface $A = E_0 \times E_2$ equipped with the product polarization, we have the pairing [33, Section 23, p.228]

$$e_A : A[3^b] \times A[3^b] \to \mu_{3^b}$$
$$e_A((P,Q),(R,S)) = e_{E_0}(P,R) \cdot e_{E_2}(Q,S)$$

where $\mu_n = \{z \in \overline{\mathbb{F}}_p^\times : z^n = 1\}$. Given (R_0, S_0) and (R_1, S_1), one may attempt to recover the masking factor γ_3' by computing the two-dimensional pairing and exploiting the following relations:

$$\begin{aligned} e_A((R_0, 0_{E_2}), (S_0, 0_{E_2})) &= e_{E_0}(R_0, S_0) \cdot e_{E_2}(0_{E_2}, 0_{E_2}) = e_{E_0}(R_0, S_0) \\ e_A((R_0, 0_{E_2}), (S_0, 0_{E_2}))^{\deg \Phi} &= e_A(\Phi(R_0, 0_{E_2}), \Phi(S_0, 0_{E_2})) \\ e_A(\Phi(R_0, 0_{E_2}), \Phi(S_0, 0_{E_2})) &= e_{E_1}(R_1, S_1)^{\gamma_3'^{-2}} \cdot e_{E_0}(R_0, S_0)^{\deg \phi_2}. \end{aligned}$$

By solving the discrete logarithm $e_{E_0}(R_0, S_0)^x = e_{E_1}(R_1, S_1)$ in the smooth-order of 3^b, one can obtain

$$x = \gamma_3'^2 \cdot (2^a - \deg \phi_2) \pmod{3^b}.$$

However, since $\deg \phi_2$ is randomly chosen, γ_3' is hidden, and the distribution of x is uniformly random and independent of γ_3'. As noted in [6, Section 6.1], by solving the discrete logarithm $e_{E_0}(R_0, S_0)^y = e_{E_2}(R_2, S_2)$, we also obtain

$$y = \gamma_3^2 \cdot \deg \phi_2 \cdot (2^a - \deg \phi_2) \pmod{3^b}.$$

Thus, we have $y/x \equiv \frac{\gamma_3^2}{\gamma_3'^2} \cdot \deg \phi_2 \pmod{3^b}$. This reveals the quadratic residuosity of q and $2^a - q$ in modulus 3^b. However, in the settings of both POKÉ and INKE, the condition $\gcd(q(2^a - q), 3) = 1$ implies that $q \equiv 2 \pmod 3$ when a is even. Under this constraint, both q and $2^a - q$ are necessarily quadratic non-residues modulo 3^b. Consequently, the pairing does not reveal any additional information about q, since the quadratic residuosity of both values is already fixed. In contrast, POKÉ imposes an additional constraint $\gcd(q(2^a - q), 5) = 1$, which reduces the search space for q by a factor of $3/5$. As a result, INKE leaks less information than POKÉ, with the difference corresponding to approximately 0.74-bit of security.

Exploiting the torsion points on the curve E_{A_1} According to the recent work of Moriya [31], there exist weak-key attacks against POKÉ-like schemes. POKÉ remains secure against this attack because the hidden degree of the receiver's secret is $q(2^a - q)$, which is approximately of the same magnitude as 3^b, the order of the disclosed torsion points.

For INKE, however, the intermediate curve E_{A_1} and the torsion points R_{A_1}, S_{A_1} on E_{A_1} are revealed. Thus, the hidden degrees are q and $(2^a - q)$, which are much smaller than the hidden degree in POKÉ for the same parameters a and q. Consequently, to achieve security against this attack, we require the parameter choice $a \approx 2\lambda$.

The attack exploits the known value $x = \gamma_3^2 \cdot \deg \phi_1$, which can be obtained by computing a pairing followed by a discrete logarithm. By [31, Theorem 11], if the degree q satisfies $3^{2b'} - q = r \cdot c^2$ for some $b' \leq b$, some integer c, and a fixed integer r, then there exists a polynomial-time algorithm that recovers the isogeny ϕ_1. According to [31, Corollary 15], the heuristic probability that a randomly generated secret key of INKE has weak degree is approximately $\frac{1}{2^{a/2}}$. Therefore, to achieve λ-bit security, one must choose $a \approx 2\lambda$.

4.2 Analysis of Sender's Security

The following problem is our modified version of [6, Problem 12] for the security against key-recovery attacks to the sender's isogenies.

Definition 3. *For an algorithm $\mathcal{A}$, we define the advantage of finding an efficient representation of ψ, given the output of* C-INKE(1^λ) *(Experiment 1) as* $\mathsf{Adv}^{\mathsf{Send}}(\mathcal{A})$.

Our sender's security is identical to that of POKÉ except that the 5^c-torsion information is not revealed, which means that the sender problem in POKÉ reduces to the sender problem of INKE. In POKÉ, the authors formalized the relationship between their sender problem and the CIST^2 problem [34, Problem 3] by showing that both have a reduction from an instance of the CIST problem (Definition 4). In the same spirit, we show a reduction from the same problem to our sender problem. Here, we formalize Problem 13 from the POKÉ paper [6] and refer to it as the *stronger version of the sender's security* (Str-Send), as it captures the sender's security even in the stronger setting where the receiver's secret key is given.

Definition 4 (Problem 13 in [6]). *For an algorithm $\mathcal{A}$, we define the advantage of finding an efficient representation of ψ, given the output of Str-Send (Experiment 3) as* $\mathsf{Adv}^{\mathsf{Str\text{-}Send}}(\mathcal{A})$.

Theorem 3 (Theorem 14 in [6]). *For any PPT adversary $\mathcal{A}$ for the Send problem, there exists a PPT algorithm $\mathcal{B}$ such that*

$$\mathsf{Adv}^{\mathsf{Send}}(\mathcal{A}) \leq \mathsf{Adv}^{\mathsf{Str\text{-}Send}}(\mathcal{B})$$

The proof can be found in Appendix C.2.

Experiment 3: (Strong) Sender's Security

Str-Send(1^λ)

1. Choose a random integer $q \in [0, 2^a)$ such that $q \equiv 5 \pmod 6$.
2. Sample a pair of isogenies $(\phi_1, \phi_2) \leftarrow \mathsf{Gen22Isog}(p, q)$ and let $\phi = \phi_2 \circ \phi_1 : E_0 \to E_A$.
3. Choose a uniformly random cyclic isogeny $\psi : E_0 \to E_B$ of degree 3^b.
4. Compute $\psi'' = \phi_* \psi : E_A \to E_{AB}$.
5. Choose a uniformly random scalar $\omega_2 \in \mathbb{Z}_{2^a}^\times$.
6. Compute a masked basis $(P_B, Q_B) = ([\omega_2]\psi(P_0), [1/\omega_2]\psi(Q_0))$.
7. Return $E_0, (P_0, Q_0), \phi, E_B, (P_B, Q_B), \phi' = \psi_* \phi$.

As noted in the POKÉ paper [6], the Str-Send problem is very similar to the CIST problem. The given isogeny and its pushforward are represented by a kernel point of ϕ and its evaluated point, up to scalar. The Str-Send problem can be thought of as an instance of the CIST problem where the order of torsion points is $q(2^a - q)$.

4.3 Security Proof of INKE in the Algebraic Isogeny Model

In INKE, assuming both the receiver's security (Definition 2) and the strong sender's security (Definition 4), we establish that the C-INKE problem is secure against algebraic adversaries by using a quantum reduction. Building on the recent security framework of Aardal, Basso and Riepel [2], we apply their general DLOG-to-CDH reduction proof [2, Theorem 6.7] to our setting.

However, in the case of that the starting curve E_0 is fixed, the DLOG-to-CDH reduction of Aardal, Basso, and Riepel does not yield a fully satisfactory reduction solely based on sender and receiver's key recovery assumptions. It requires the hardness assumption of computing a non-scalar endomorphism on a party's image curve.

Recent work by Mingjie Chen and Christophe Petit [13] shows that, given a full-rank suborder of the endomorphism ring of a supersingular curve, there exists a quantum polynomial time algorithm that recovers the entire endomorphism ring. Moreover, they prove that, given the endomorphism ring of a domain curve together with an isogeny path, there is a quantum polynomial time algorithm that computes the endomorphism ring of the codomain curve (generalization of the result in [12]).

By combining these as algorithmic tools, we show that the C-INKE problem admits a quantum reduction that relies solely on the sender's and receiver's security assumptions.

Prior to the reduction proof, we first clarify what constitutes an efficient representation of an isogeny. In particular, we adopt the notion of an *efficient isogeny representation* introduced in [2].

Definition 5 (Efficient isogeny representation [2]). *An efficient isogeny representation for an isogeny $\varphi : E \to E'$, is a string s_φ of size $O(\mathsf{poly}(\log(p + \deg\varphi))$ such that there exists a pair of polynomial time algorithms:*

- $\mathcal{E}.\mathsf{valid}(s_\varphi)$*: On input a string s_φ, it outputs $\perp$ or a triple (E, E', d). In the latter case, E and E' are elliptic curves defined over $\mathbb{F}_p$ and s_φ represents an isogeny $\varphi : E \to E'$ of degree d.*
- $\mathcal{E}.\mathsf{eval}(s_\varphi, P)$*: On input a string s_φ and a point $P \in E(\mathbb{F}_{q^k})$, it outputs the image point $\varphi(P) \in E'(\mathbb{F}_{q^k})$ if $\mathcal{E}.\mathsf{valid}(E) = (E, E', d)$, otherwise the output is undefined.*

In the case that $\mathcal{E}.\mathsf{valid}(s_\varphi) \neq \perp$, we say that s_φ is an efficient representation of φ (with respect to $\mathcal{E}$).

Using the notion of the efficient isogeny representation, the algebraic isogeny model is defined as follows.

Definition 6. *In the algebraic isogeny model, an algorithm $\mathcal{A}$ said to be algebraic as follows. Let $\mathcal{A}$ gets as input the curves $E_1, \ldots, E_n$. If $\mathcal{A}$ outputs a curve E, it also outputs an efficient representation $(s_{\varphi_{\mathsf{expl}}}, \mathcal{E})$ of an isogeny $\varphi_{\mathsf{expl}} : E_i \to E$ for some $1 \leq i \leq n$ or $\varphi_{\mathsf{expl}} : E_{\mathsf{End}} \to E$ where E_{End} is a curve with known endomorphism ring. We call the isogeny φ_{expl} the explanation for E.*

Within the AIM security framework, we formulate our main theorem (Theorem 4) as follows.

Theorem 4. *For any PPT algebraic adversary $\mathcal{A}$ that solves the C-INKE problem, there exists a quantum adversary $\mathcal{B}$ that solves the Recv problem or the Str-Send problem with constant probability in expected running time $O(1/\varepsilon)$ where $\varepsilon = \mathsf{Adv}^{\mathsf{C\text{-}INKE}}(\mathcal{A})$.*

To prove Theorem 4, we use the following useful quantum reductions between isogeny hard problems proved in [13].

Theorem 5 (IsERP, Theorem 3 in [13](heuristic)). *Let $\tilde{E}, E$ be two supersingular elliptic curves over $\mathbb{F}_{p^2}$ and let $\varphi : \tilde{E} \to E$ be an isogeny of degree N for some integer N. Given $\mathrm{End}(\tilde{E})$ and an efficient isogeny representation for φ, there exists a quantum polynomial algorithm that computes $\mathrm{End}(E)$ in expected polynomial time in $\log p$ and $\log N$.*

Theorem 6 (SubOrderEndRing, Theorem 1 in [13](heuristic)). *Let E be a supersingular elliptic curve over $\mathbb{F}_{p^2}$. Given a full rank suborder $\mathcal{R}_E \subseteq \mathrm{End}(E)$ of discriminant $\Delta_\mathcal{R}$, there exists a quantum algorithm that computes $\mathrm{End}(E)$ in expected polynomial time in $\log p$ and $\log(\sqrt{\Delta_\mathcal{R}}/p)$.*

Remark 1. The heuristics involved in Theorem 5 and 6 are from the plausible heuristics of subroutines RepresentInteger and GetStrongApproximation.

We introduce our algorithms which are used as subroutines in the reduction algorithm of Theorem 4. The details of the algorithms are described in Appendix A.

- $\mathsf{ExtractSend}((s_\varphi, \mathcal{E}), x)$ (Algorithm 9): takes as input a C-INKE instance x and an efficient representation $(s_\varphi, \mathcal{E})$ of an isogeny $\varphi : E_{A_1} \to E_{A_1 B}$, where E_{A_1} and $E_{A_1 B}$ are curves associated with the instance x. It outputs the sender's secret isogeny ψ corresponding to x. If φ is not extractable, it aborts.
- $\mathsf{ExtractRecv}((s_\varphi, \mathcal{E}), x)$ (Algorithm 10): takes as input a Recv instance x and an efficient representation $(s_\varphi, \mathcal{E})$ of an isogeny $\varphi : E \to E_{A_1 B}$, where $E_{A_1 B}$ is the codomain curve appearing in a C-INKE instance containing x. It outputs the receiver's secret isogeny $\phi : E_0 \to E_A$ corresponding to the instance x. The algorithm uses a factorization oracle as a subroutine: it factors $\deg \varphi$ and recovers the degree of the receiver's secret isogeny ϕ from this factorization. If φ is not extractable, it aborts.
- $\mathsf{ExtendRecv}(x)$ (Algorithm 11): takes as input a Recv instance x, and outputs three cyclic 3^b-isogenies (ψ, ψ', ψ'') together with an associated C-INKE instance y containing the instance x. This algorithm constructs a new C-INKE instance y for which the sender's secret isogeny ψ is known.
- $\mathsf{StrSendToCINKE}(x)$ (Algorithm 12): takes as input a Str-Send instance x, and outputs two C-INKE instances y_1 and y_2 together with associated isogenies $(\phi'_1, \phi'_2, \phi'_3, \phi'_4)$ satisfying

  $$\phi'_2 \circ \phi'_1 = \phi'_4 \circ \phi'_3 = \phi',$$

  where $\phi' : E_B \to E_{AB}$ is the isogeny from the instance x. Moreover,

  $$\deg \phi'_1 = \deg \phi'_4, \qquad \deg \phi'_2 = \deg \phi'_3,$$

  and

  $$\deg \phi'_1 + \deg \phi'_2 = \deg \phi'_3 + \deg \phi'_4 = 2^a.$$

  The instance y_1 is associated with the pair (ϕ'_1, ϕ'_2), while the instance y_2 is associated with the pair (ϕ'_3, ϕ'_4). This algorithm constructs new C-INKE instances y_1, y_2 for which the receiver's secret isogeny ϕ' is known.

The algorithms ExtractSend and ExtractRecv are used to extract the desired secret isogenies from the given isogeny explanations. The ExtendRecv algorithm is used to collect distinct non-scalar endomorphisms of E_{A_1} from a given Recv instance x. Similarly, the StrSendToCINKE algorithm is used to collect distinct non-scalar endomorphisms of E_B from a given Str-Send instance x.

Proof (Proof of Theorem 4). Our proof follows the proof strategy of [2, Theorem 6.7]. Let $\mathcal{A}$ be an adversary against C-INKE that outputs a curve $E_{A_1 B}$. Since $\mathcal{A}$ is algebraic, it must provide an explanation for its output in the form of one of the following isogenies:

1. an isogeny $\varphi_{\mathsf{expl}} : E_{\mathsf{end}} \to E_{A_1 B}$,
2. an isogeny $\varphi_{\mathsf{expl}} : E_{A_1} \to E_{A_1 B}$,
3. an isogeny $\varphi_{\mathsf{expl}} : E_A \to E_{A_1 B}$,
4. an isogeny $\varphi_{\mathsf{expl}} : E_B \to E_{A_1 B}$,
5. an isogeny $\varphi_{\mathsf{expl}} : E_{AB} \to E_{A_1 B}$,

where E_{end} denotes a curve whose endomorphism ring is known. In particular, the starting curve E_0 is included as a special case of E_{end}.

We construct an adversary $\mathcal{B}$ (Algorithm 15 in Appendix B) that attempts to solve either the Recv problem or the Str-Send problem by guessing the type of explanation output by $\mathcal{A}$. If the guess is incorrect, $\mathcal{B}$ aborts.

The adversary $\mathcal{B}$ maintains two tables $\mathcal{T}_{A_1}$ and $\mathcal{T}_B$ storing endomorphisms of the curve E_{A_1} and E_B, respectively. $\mathcal{B}$ also maintains a state variable σ_{known} representing an isogeny path from E_{A_1} to E_A. The algorithm proceeds as follows.

- **Case 1.** ($\varphi_{\mathsf{expl}} : E_{\mathsf{end}} \to E_{A_1B}$) If the explanation isogeny is of the form $\varphi_{\mathsf{expl}} : E_{\mathsf{end}} \to E_{A_1B}$, then $\mathcal{B}$ solves the Recv problem as follows. From an instance $x \leftarrow \mathsf{Recv}(1^\lambda)$, $\mathcal{B}$ computes $(\psi, \psi', \psi'', y) \leftarrow \mathsf{ExtendRecv}(x)$. $\mathcal{B}$ runs $\mathcal{A}$ with an input y. If $\mathcal{A}$ solves the C-INKE problem, it outputs E_{A_1B} and an explanation φ_{expl}. Since $\mathcal{B}$ knows an isogeny path $\widehat{\psi'} \circ \varphi_{\mathsf{expl}} : E_{\mathsf{end}} \to E_{A_1}$, by Theorem 5, $\mathcal{B}$ can compute $\mathrm{End}(E_{A_1})$ from $\widehat{\psi'} \circ \varphi_{\mathsf{expl}}$. By Lemma 4.2, $\mathcal{B}$ can solve the Recv problem from $\mathrm{End}(E_{A_1})$.
- **Case 2.** ($\varphi_{\mathsf{expl}} : E_{A_1} \to E_{A_1B}$) If the explanation isogeny is of the form $\varphi_{\mathsf{expl}} : E_{A_1} \to E_{A_1B}$, then $\mathcal{B}$ solves the Recv problem or the Str-Send problem. Given a Str-Send instance $y \leftarrow \mathsf{Str\text{-}Send}(1^\lambda)$, $\mathcal{B}$ first computes

$$(y_1, y_2), _ \leftarrow \mathsf{StrSendToCINKE}(y).$$

  Case 2-1. If $\mathsf{ExtractSend}((s_{\varphi_{\mathsf{expl}}}, \mathcal{E}), y_1)$ does not abort, $\mathcal{B}$ solves the Str-Send problem. Then, $\mathcal{B}$ computes

$$\begin{aligned}(E_{A_1B}, (s_{\varphi_{\mathsf{expl}}}, \mathcal{E})) &\leftarrow \mathcal{A}(y_1)\\ \psi &\leftarrow \mathsf{ExtractSend}((s_{\varphi_{\mathsf{expl}}}, \mathcal{E}), y_1).\end{aligned}$$

  $\mathcal{B}$ outputs ψ as a solution of the Str-Send problem of y.
  Case 2-2. If $\mathsf{ExtractSend}((s_{\varphi_{\mathsf{expl}}}, \mathcal{E}), y_1)$ aborts, then $\mathcal{B}$ uses Lemma 4.3 to obtain a full-rank suborder of $\mathrm{End}(E_{A_1})$. Given an instance $x \leftarrow \mathsf{Recv}(1^\lambda)$, $\mathcal{B}$ computes

$$\begin{aligned}(\psi_1, \psi_1', \psi_1'', y_3) &\leftarrow \mathsf{ExtendRecv}(x)\\ (\psi_2, \psi_2', \psi_2'', y_4) &\leftarrow \mathsf{ExtendRecv}(x).\end{aligned}$$

  By running

$$\begin{aligned}(E_{1,A_1B}, \varphi_{1,\mathsf{expl}}) &\leftarrow \mathcal{A}(y_3)\\ (E_{2,A_1B}, \varphi_{2,\mathsf{expl}}) &\leftarrow \mathcal{A}(y_4),\end{aligned}$$

  $\mathcal{B}$ obtains a subring of $\mathrm{End}(E_{A_1})$ generated by

$$\{1, \widehat{\varphi}_{1,\mathsf{expl}} \circ \psi_1', \widehat{\varphi}_{2,\mathsf{expl}} \circ \psi_2'\}.$$

  By Lemma 4.3, this subring has rank 4 with probability at least $1/2$. Using Theorem 6, $\mathcal{B}$ can recover $\mathrm{End}(E_{A_1})$ from the resulting full-rank suborder. Then, by Lemma 4.2, $\mathcal{B}$ can solve the Recv problem from $\mathrm{End}(E_{A_1})$.

In the actual Algorithm 15, the adversary $\mathcal{B}$ repeats the execution of $\mathcal{A}$ and stores the non-scalar endomorphisms into the table $\mathcal{T}_{A_1}$ approximately $2/\varepsilon$ times in order to avoid the quadratic overhead that would arise from waiting for multiple consecutive successful outputs of $\mathcal{A}$.

– **Case 3.** ($\varphi_{\mathsf{expl}} : E_A \to E_{A_1B}$) If the explanation isogeny is of the form $\varphi_{\mathsf{expl}} : E_A \to E_{A_1B}$, then $\mathcal{B}$ solves the Str-Send problem or the Recv problem by using the similar way of Case 2. If $\sigma_{\mathsf{known}} = \perp$, i.e. there is no known isogeny path σ_{known} from E_{A_1} to E_A, store the path $\sigma_{\mathsf{known}} = \widehat{\varphi}_{\mathsf{expl}} \circ \psi' : E_{A_1} \to E_A$ and abort. Otherwise, $\mathcal{B}$ does the following.
Given a Str-Send instance $y \leftarrow \mathsf{Str\text{-}Send}(1^\lambda)$, $\mathcal{B}$ first computes

$$(y_1, y_2), _ \leftarrow \mathsf{StrSendToCINKE}(y).$$

Case 3-1. If $\mathsf{ExtractSend}((s_{\varphi_{\mathsf{expl}} \circ \sigma_{\mathsf{known}}}, \mathcal{E}), y_1)$ does not abort, $\mathcal{B}$ solves the Send problem. Given an instance $x \leftarrow \mathsf{C\text{-}INKE}(1^\lambda)$, $\mathcal{B}$ runs

$$(E_{A_1B}, (s_{\varphi_{\mathsf{expl}} \circ \sigma_{\mathsf{known}}}, \mathcal{E})) \leftarrow \mathcal{A}(y_1)$$
$$\psi \leftarrow \mathsf{ExtractSend}((s_{\varphi_{\mathsf{expl}} \circ \sigma_{\mathsf{known}}}, \mathcal{E}), y_1).$$

$\mathcal{B}$ outputs ψ as a solution of the Send problem. By Theorem 3, $\mathcal{B}$ solves the Str-Send problem.
Case 3-2. If $\mathsf{ExtractSend}((s_{\varphi_{\mathsf{expl}} \circ \sigma_{\mathsf{known}}}, \mathcal{E}), y_1)$ aborts, then $\mathcal{B}$ solves the Recv problem. Given an instance $x \leftarrow \mathsf{Recv}(1^\lambda)$, $\mathcal{B}$ computes

$$(\psi_1, \psi_1', \psi_1'', y_3) \leftarrow \mathsf{ExtendRecv}(x)$$
$$(\psi_2, \psi_2', \psi_2'', y_4) \leftarrow \mathsf{ExtendRecv}(x).$$

By running

$$(E_{1,A_1B}, \varphi_{1,\mathsf{expl}}) \leftarrow \mathcal{A}(y_3)$$
$$(E_{2,A_1B}, \varphi_{2,\mathsf{expl}}) \leftarrow \mathcal{A}(y_4),$$

$\mathcal{B}$ obtains a subring of $\mathrm{End}(E_{A_1})$ generated by $\{1, \widehat{\sigma}_{\mathsf{known}} \circ \sigma_1, \widehat{\sigma}_{\mathsf{known}} \circ \sigma_2\}$ where $\sigma_i = \widehat{\varphi}_{i,\mathsf{expl}} \circ \psi_i' : E_{A_1} \to E_A$. By Lemma 4.3, with the probability at least $1/2$, the subring has rank 4. By using Theorem 6, $\mathcal{B}$ can compute $\mathrm{End}(E_{A_1})$ from the full-rank suborder. By Lemma 4.2, $\mathcal{B}$ can solve the Recv problem from the $\mathrm{End}(E_{A_1})$. As in Case 2-2, in Algorithm 15 in Appendix B, $\mathcal{B}$ runs $\mathcal{A}$ approximately $3/\varepsilon$ times, and stores the resulting non-scalar endomorphisms in the table $\mathcal{T}_{A_1}$ to avoid cubical overhead arises from waiting for multiple consecutive success of $\mathcal{A}$.

– **Case 4.** ($\varphi_{\mathsf{expl}} : E_B \to E_{A_1B}$) If the explanation isogeny is of the form $\varphi_{\mathsf{expl}} : E_B \to E_{A_1B}$, then $\mathcal{B}$ will solve the Recv problem as follows. Given an instance $x \leftarrow \mathsf{Recv}(1^\lambda)$, $\mathcal{B}$ computes $(\psi, \psi', \psi'', y) \leftarrow \mathsf{ExtendRecv}(x)$. By running $(E_{A_1B}, \varphi_{\mathsf{expl}}) \leftarrow \mathcal{A}(x)$, $\mathcal{B}$ obtains an isogeny path $\sigma = \widehat{\psi'} \circ \varphi_{\mathsf{expl}} \circ \psi : E_0 \to E_{A_1}$. By Theorem 5, $\mathcal{B}$ obtains $\mathrm{End}(E_{A_1})$ from the isogeny path σ. By Lemma 4.2, $\mathcal{B}$ computes a solution of the Recv problem from the endomorphism ring $\mathrm{End}(E_{A_1})$.

– **Case 5.** ($\varphi_{\mathsf{expl}} : E_{AB} \to E_{A_1B}$) If the explanation isogeny is of the form $\varphi_{\mathsf{expl}} : E_{AB} \to E_{A_1B}$, $\mathcal{B}$ solves the Str-Send problem or the Recv problem. Let $x \leftarrow \mathsf{Recv}(1^\lambda)$.
Case 5-1. If $\mathsf{ExtractRecv}((s_{\varphi_{\mathsf{expl}}}, \mathcal{E}), x)$ does not abort, $\mathcal{B}$ solves the Recv problem. From the Recv instance x, $\mathcal{B}$ computes $y \leftarrow \mathsf{ExtendRecv}(x)$, and runs

$$(E_{A_1B}, (s_{\varphi_{\mathsf{expl}}}, \mathcal{E})) \leftarrow \mathcal{A}(y)$$
$$\psi \leftarrow \mathsf{ExtractRecv}((s_{\varphi_{\mathsf{expl}}}, \mathcal{E}), x).$$

$\mathcal{B}$ outputs ψ as a solution of the Recv problem of x.
Case 5-2. If $\mathsf{ExtractRecv}((s_{\varphi_{\mathsf{expl}}}, \mathcal{E}), x)$ aborts, then $\mathcal{B}$ solves the Str-Send problem. Given an instance $x = (E_0, (P_0, Q_0), \phi, E_B, (P_B, Q_B), \phi') \leftarrow \mathsf{Str\text{-}Send}(1^\lambda)$, $\mathcal{B}$ computes

$$(y_1, y_2), (\phi'_1, \phi'_2, \phi'_3, \phi'_4) \leftarrow \mathsf{StrSendToCINKE}(x).$$

$\mathcal{B}$ runs

$$(E_{12,A_1B}, \varphi_{12,\mathsf{expl}}) \leftarrow \mathcal{A}(y_1)$$
$$(E_{34,A_1B}, \varphi_{34,\mathsf{expl}}) \leftarrow \mathcal{A}(y_2),$$

$\mathcal{B}$ obtains a subring of $\mathrm{End}(E_B)$ generated by $\{1, \widehat{\phi'_1} \circ \varphi_{12,\mathsf{expl}} \circ \phi', \widehat{\phi'_3} \circ \varphi_{34,\mathsf{expl}} \circ \phi'\}$. By Lemma 4.4, with the probability at least $2/3$, the subring has rank 4. By using Theorem 6, $\mathcal{B}$ can compute $\mathrm{End}(E_B)$ from the full-rank suborder. By using Lemma 4.1, $\mathcal{B}$ can recover the isogeny $\psi : E_0 \to E_B$ by using $\mathrm{End}(E_B)$ and $\mathrm{End}(E_0)$. In the actual Algorithm 15, the adversary $\mathcal{B}$ repeats the execution of $\mathcal{A}$ and stores the non-scalar endomorphisms into the table $\mathcal{T}_B$ approximately $2/\varepsilon$ times in order to avoid the quadratic overhead that would arise from waiting for multiple consecutive successful outputs of $\mathcal{A}$.

As the guess of $\mathcal{B}$ is independent of $\mathcal{A}$, the security loss is a constant factor 8.

Note that in order to remain the running time of $\mathcal{B}$ linear in $1/\varepsilon$, the algorithm $\mathcal{B}$ is designed to repeat the overall procedure multiple times while storing any newly obtained non-scalar endomorphisms of E_{A_1} and E_B in $\mathcal{T}_{A_1}$ and $\mathcal{T}_B$, respectively. In Case 2-2, a non-scalar endomorphism can be obtained directly and stored in the table. In contrast, in Cases 3-2, the algorithm only recovers an isogeny path from E_{A_1} to E_A. In these cases, once such a path is obtained, it is combined with a previously known path σ_{known} to derive a non-scalar endomorphism, which is then stored. □

Lemma 4.1 (Heuristic). *Let $x \leftarrow$ C-INKE(1^λ). Given an efficient representation of a basis of* $\mathrm{End}(E_B)$*, there exists an algorithm that solves the* Send *problem of x in expected polynomial time in λ.*

The proof can be found in Appendix C.3.

Lemma 4.2 (Heuristic). *Let $x \leftarrow \mathsf{Recv}(1^\lambda)$. Given an efficient representation of a basis of $\mathrm{End}(E_{A_1})$, there exists an algorithm that solves the Recv problem of x in expected polynomial time in λ.*

The proof can be found in Appendix C.4.

Remark 2. The heuristics used in Lemmas 4.1 and 4.2 are based on the Gaussian heuristic applied to the rank-4 lattice of isogenies between the given supersingular elliptic curves. A justification for these heuristics is provided in Appendix C.7.

Lemma 4.3. *Let $x \leftarrow \mathsf{Recv}(1^\lambda)$. Let $(\psi_1, \psi_1', \psi_1'', y_1)$ and $(\psi_2, \psi_2', \psi_2'', y_2)$ be outputs from $\mathsf{ExtendRecv}(x)$ by running it twice. Let $\varphi_i : E_{A_1} \to E_i$ be isogenies of arbitrary degree $(i = 1, 2)$. Let $\sigma_i = \widehat{\varphi}_i \circ \psi_i' \in \mathrm{End}(E_{A_1})$. If both $\mathsf{ExtractSend}(\varphi_i, y_i)$ abort, the probability that the subring of $\mathrm{End}(E_{A_1})$ generated by $\{1, \sigma_1, \sigma_2\}$ has rank 4 is at least $1/2$.*

Proof (Proof sketch). The idea is to exploit the fact that the isogenies ψ_i are chosen randomly in the $\mathsf{ExtendRecv}$ algorithm, while the unextractable isogenies φ_i cannot overlap with the ψ_i. Consequently, the endomorphisms σ_1 and σ_2 are constrained by the independently chosen isogenies ψ_1 and ψ_2, respectively. This implies that the endomorphisms σ_1, σ_2, and the identity map are linearly independent with constant probability.

The complete proof can be found in Appendix C.5.

Lemma 4.4 (Heuristic). *In Case 5-2 of the proof of Theorem 4, the $\mathbb{Z}$-algebra generated by*

$$\{1,\ \widehat{\phi}_1' \circ \varphi_{12,\mathsf{expl}} \circ \phi',\ \widehat{\phi}_3' \circ \varphi_{34,\mathsf{expl}} \circ \phi'\}$$

has rank 4 with probability at least $\frac{2}{3}$.

Proof (Proof sketch). The main idea is to exploit the fact that, for an unextractable isogeny φ, there exists a prime t such that $t \mid q$ and $t \nmid \deg \varphi$. Since the two degrees q and $2^a - q$ are coprime, and the isogenies ϕ_1 and ϕ_3 are heuristically chosen uniformly at random in the $\mathsf{Gen22Isog}$ algorithm, the given endomorphisms are linearly independent on a t^ℓ-torsion subgroup for some ℓ with constant probability.

The complete proof can be found in Appendix C.6.

Remark 3. The heuristic used in Lemma 4.4 relies on the assumption that the output of the $\mathsf{FullRepresentInteger}$ algorithm used in $\mathsf{Gen22Isog}$ within $\mathsf{Str\text{-}Send}$ behaves randomly.

5 Implementation Results

This section presents the implementation results of INKE. We first present the parameters of INKE for NIST security levels 1, 3, and 5, and compare their sizes with those of POKÉ. Subsequently, we provide a performance comparison between the two schemes.

5.1 Parameter selection

For a security parameter λ, the parameter a is chosen so that $a \approx 2\lambda$,[5] since the secret isogeny degree q is sampled from $[0, 2^a)$ and must resist brute-force search attacks. The parameter b is chosen so that $3^b \approx 2^{2\lambda}$, in order to resist meet-in-the-middle attacks on the 3^b-isogeny. Consequently, the resulting prime has size approximately 4λ bits. In comparison, the prime size required for POKÉ is approximately $10\lambda/3$ bits, so our scheme requires the larger prime size by roughly λ bits. We provide the INKE primes for the NIST-I, NIST-III, and NIST-V security levels below.

$$\begin{aligned}
\textbf{NIST-I}: &\quad p = 2^{257} \cdot 3^{163} \cdot 11 - 1,\ \log_2(p) = 519 \\
\textbf{NIST-III}: &\quad p = 2^{386} \cdot 3^{244} \cdot 23 - 1,\ \log_2(p) = 778 \\
\textbf{NIST-V}: &\quad p = 2^{512} \cdot 3^{324} \cdot 11 \cdot 47 - 1,\ \log_2(p) = 1035
\end{aligned}$$

Based on this parameters, Table 2 compares the public-key and ciphertext sizes of INKE, POKÉ and PIKE at the 128-, 192-, and 256-bit security levels.

Table 1: Comparison of components of public key, secret key, and ciphertext between POKÉ and INKE.

	POKÉ	INKE
pk	$E_A, P_A, Q_A, R_A, S_A, X_A, Y_A$	$E_A, P_A, Q_A, R_A, S_A, E_{A_1}, R_{A_1}, S_{A_1}$
sk	$q, \alpha_2, \beta_2, \delta_5$	q, α_2, β_2
ct	$(E_B, P_B, Q_B), (E_{AB}, P_{AB}, Q_{AB}), \mathsf{ct}'$	$(E_B, P_B, Q_B), (E_{AB}, P_{AB}, Q_{AB}), \mathsf{ct}'$

As noted by the authors of POKÉ in [6], points of coprime order can be reduced into a single representative point by summing them. Since a POKÉ public key consists of one elliptic curve (of size $2\log p$) and three sets of bases of coprime orders on the same curve, it can be represented using $6\log p$ bits. The ciphertext, which consists of two elliptic curves and two sets of bases, can be represented using $12\log p$ bits.

Additional compression is possible by expressing the basis with respect to a deterministic basis and using pairings. Under this compression, the public key can be reduced to $5\log p$ bits, and the ciphertext to $4\log p + 6a$ bits, where a is the exponent of 2 in $p+1$.

Remark 4. The ciphertext size of POKÉ is in fact larger. In POKÉ encryption, the points X_B and Y_B are masked by a random invertible matrix. As mentioned

[5] This change reflects the attack proposed in [31], which requires increasing the parameter a from λ to 2λ in order to maintain the targeted security level.

Table 2: Comparison of public key and ciphertext sizes of INKE, POKÉ and PIKE on an Intel i7-9700 CPU. The sizes are measured in bytes.

Metric	NIST I			NIST III			NIST V		
	INKE	POKÉ	PIKE	INKE	POKÉ	PIKE	INKE	POKÉ	PIKE
$\log_2 p$	519	431	384	778	648	576	1035	863	770
pk size (B)	779	324	480	1167	486	720	1553	648	963
ct size (B)	779	648	576	1167	972	864	1553	1296	1156
$\mathsf{pk}_{\mathsf{cmp}}$ size (B)	551	270	272	826	405	408	1099	540	545
$\mathsf{ct}_{\mathsf{cmp}}$ size (B)	452	312	400	677	468	600	902	624	802

in the POKÉ paper [6], one could reduce the ciphertext size by choosing X_B, Y_B as a canonical basis of $E_B[5^c]$ and omitting them from the ciphertext.

However, this optimization increases the encryption cost. In particular, it requires solving a discrete logarithm problem to recover the masking matrix D_5 satisfying

$$[X_B, Y_B]^T = D_5[\psi(X_0), \psi(Y_0)]^T,$$

and then computing

$$[X_{AB}, Y_{AB}]^T = D_5[\psi'(X_A), \psi'(Y_A)]^T.$$

Thus, there is a trade-off between ciphertext size and encryption efficiency: using a canonical basis reduces the ciphertext size, but introduces additional overhead from the discrete logarithm computation. For a fair comparison, we do not apply this optimization in our performance evaluation.

5.2 Performance analysis

For performance analysis, we implemented the uncompressed versions of POKÉ and INKE in C by modifying the SQIsign2D reference code[6], which implements the $(2, 2)$-isogeny computation between abelian surfaces proposed in [15]. The implementation relies on GMP[7] for multi-precision integer and quaternion arithmetic, while finite field arithmetic over $\mathbb{F}_p$ is generated using Fiat-Crypto [17]. We also optimized the PIKE reference implementation[8] from [8] for a fair comparison.

Benchmarks were performed on an Intel i7-9700 Coffee Lake CPU at 3 GHz running Ubuntu 24.04.3 LTS with the clang 20.1.8 compiler. Intel Turbo Boost, EIST, and C-States were disabled during measurements. Table 3 reports the average timings over 100 runs. Our C implementation incorporates the following optimizations:

[6] https://github.com/SQISign/sqisign2d-west-ac24
[7] https://gmplib.org/
[8] https://github.com/Kaizhan-Lin/PIKE-C-Implementation

- Optimized finite field arithmetics based on [17].
- Precomputation of endomorphism actions on $E_0 : y^2 = x^3 + x$.
- Use of optimal strategies for 3-, 5-, and $(2, 2)$- isogenies.
- Optimized discrete logarithm computations using cubical arithmetic for pairings [38], [36].

The comparison between INKE and POKÉ is shown in Table 3.

Table 3: Comparison of C-implementations of POKÉ, PIKE and INKE on an Intel i7-9700 CPU. The runtimes are reported in Mcycles.

Level	**INKE (Mcycles)**			**POKÉ (Mcycles)**			**PIKE (Mcycles)**		
	KeyGen	Enc	Dec	KeyGen	Enc	Dec	KeyGen	Enc	Dec
NIST-I	702	162	187	359	83	122	372	71	65
NIST-III	2113	473	548	1358	319	469	1271	250	222
NIST-V	5057	1076	1253	2950	685	997	2760	543	480

Table 4: Comparison of the decapsulation (Algorithm 16, 17 and 18 in Appendix E) of INKE, POKÉ and PIKE. Timings are the average of 100 executions on an Intel i7-9700 CPU.

Level	**Decapsulation (Mcycles)**		
	INKE	POKÉ	PIKE
NIST-I	252	169	124
NIST-III	757	644	425
NIST-V	1702	1400	927

We also provide C implementations of the FO-transformed KEM variants of INKE, POKÉ, and PIKE. In general, decapsulation requires both encryption and decryption costs. However, by adapting the simplified re-encryption technique of SIKE KEM [23], decapsulation in INKE can be performed with only one additional 3^b-isogeny computation and two point evaluations (three in the x-only setting). Table 4 describes the benchmark result. Detailed optimized decapsulation algorithms are provided in Appendix E.

Acknowledgements. We thank Gwanho Kim for insightful discussions and valuable comments on this work.

References

1. Aardal, M.A., Basso, A., De Feo, L., Patranabis, S., Wesolowski, B.: A complete security proof of sqisign. In: Tauman Kalai, Y., Kamara, S.F. (eds.) Advances in

Cryptology – CRYPTO 2025. pp. 190–222. Springer Nature Switzerland, Cham (2025). https://doi.org/10.1007/978-3-032-01887-8_7
2. Aardal, M.A., Basso, A., Riepel, D.: The algebraic isogeny model: A general model with applications to sqisign and key exchanges. Cryptology ePrint Archive (2026)
3. Basso, A., Borin, G., Rueger, R., Schaeffler, S.: CORAL faster isogeny group action for post-quantum NIKE. Cryptology ePrint Archive, Paper 2026/896 (2026), https://eprint.iacr.org/2026/896
4. Basso, A., Codogni, G., Connolly, D., De Feo, L., Fouotsa, T.B., Lido, G.M., Morrison, T., Panny, L., Patranabis, S., Wesolowski, B.: Supersingular curves you can trust. In: Hazay, C., Stam, M. (eds.) Advances in Cryptology – EUROCRYPT 2023. pp. 405–437. Springer Nature Switzerland, Cham (2023)
5. Basso, A., Dartois, P., Feo, L.D., Leroux, A., Maino, L., Pope, G., Robert, D., Wesolowski, B.: SQIsign2D–west. In: Chung, K.M., Sasaki, Y. (eds.) Advances in Cryptology – ASIACRYPT 2024. pp. 339–370. Springer Nature Singapore, Singapore (2025). https://doi.org/10.1007/978-981-96-0891-1_11
6. Basso, A., Maino, L.: Poké: A compact and efficient pke from higher-dimensional isogenies. In: Fehr, S., Fouque, P.A. (eds.) Advances in Cryptology – EUROCRYPT 2025. pp. 94–123. Springer Nature Switzerland, Cham (2025). https://doi.org/10.1007/978-3-031-91124-8_4
7. Basso, A., Maino, L., Pope, G.: FESTA: Fast encryption from supersingular torsion attacks. In: Guo, J., Steinfeld, R. (eds.) Advances in Cryptology – ASIACRYPT 2023. pp. 98–126. Springer Nature Singapore, Singapore (2023). https://doi.org/10.1007/978-981-99-8739-9_4
8. Cai, S., Chen, M., Lai, Y.F., Lin, K.: Pike: Faster isogeny-based public key encryption with pairing-assisted decryption. Cryptology ePrint Archive (2026)
9. Castryck, W., Decru, T.: An efficient key recovery attack on SIDH. In: Hazay, C., Stam, M. (eds.) Advances in Cryptology – EUROCRYPT 2023. pp. 423–447. Springer Nature Switzerland, Cham (2023). https://doi.org/10.1007/978-3-031-30589-4_15
10. Castryck, W., Lange, T., Martindale, C., Panny, L., Renes, J.: Csidh: An efficient post-quantum commutative group action. In: Advances in Cryptology – ASIACRYPT 2018: 24th International Conference on the Theory and Application of Cryptology and Information Security, Brisbane, QLD, Australia, December 2–6, 2018, Proceedings, Part III. p. 395–427. Springer-Verlag, Berlin, Heidelberg (2018). https://doi.org/10.1007/978-3-030-03332-3_15, https://doi.org/10.1007/978-3-030-03332-3_15
11. Charles, D.X., Lauter, K.E., Goren, E.Z.: Cryptographic hash functions from expander graphs. Journal of CRYPTOLOGY **22**(1), 93–113 (2009)
12. Chen, M., Imran, M., Ivanyos, G., Kutas, P., Leroux, A., Petit, C.: Hidden stabilizers, the isogeny to endomorphism ring problem and the cryptanalysis of psidh. In: Guo, J., Steinfeld, R. (eds.) Advances in Cryptology – ASIACRYPT 2023. pp. 99–130. Springer Nature Singapore, Singapore (2023). https://doi.org/10.1007/978-981-99-8727-6_4
13. Chen, M., Petit, C.: Computing the endomorphism ring of a supersingular elliptic curve from a full rank suborder. In: Fehr, S., Fouque, P.A. (eds.) Advances in Cryptology – EUROCRYPT 2025. pp. 446–474. Springer Nature Switzerland, Cham (2025). https://doi.org/10.1007/978-3-031-91095-1_16
14. Childs, A., Jao, D., Soukharev, V.: Constructing elliptic curve isogenies in quantum subexponential time. Journal of Mathematical Cryptology **8**(1), 1–29 (2014)

15. Dartois, P., Maino, L., Pope, G., Robert, D.: An algorithmic approach to (2, 2)-isogenies in the theta model and applications to isogeny-based cryptography. In: Chung, K.M., Sasaki, Y. (eds.) Advances in Cryptology – ASIACRYPT 2024. pp. 304–338. Springer Nature Singapore, Singapore (2025). https://doi.org/10.1007/978-981-96-0891-1_10
16. De Feo, L., Leroux, A., Longa, P., Wesolowski, B.: New algorithms for the deuring correspondence. In: Hazay, C., Stam, M. (eds.) Advances in Cryptology – EUROCRYPT 2023. pp. 659–690. Springer Nature Switzerland, Cham (2023)
17. Erbsen, A., Philipoom, J., Gross, J., Sloan, R., Chlipala, A.: Simple high-level code for cryptographic arithmetic: With proofs, without compromises. ACM SIGOPS Operating Systems Review **54**(1), 23–30 (2020). https://doi.org/10.1145/3421473.3421477
18. Fouque, P.A., Kirchner, P., Pornin, T., Yu, Y.: Bat: Small and fast kem over ntru lattices. IACR Transactions on Cryptographic Hardware and Embedded Systems **2022**(2), 240–265 (Feb 2022). https://doi.org/10.46586/tches.v2022.i2.240-265, https://tches.iacr.org/index.php/TCHES/article/view/9487
19. Fuchsbauer, G., Kiltz, E., Loss, J.: The algebraic group model and its applications. In: Shacham, H., Boldyreva, A. (eds.) Advances in Cryptology – CRYPTO 2018. pp. 33–62. Springer International Publishing, Cham (2018). https://doi.org/10.1007/978-3-319-96881-0_2
20. Fujisaki, E., Okamoto, T.: Secure integration of asymmetric and symmetric encryption schemes. In: Wiener, M. (ed.) Advances in Cryptology — CRYPTO' 99. pp. 537–554. Springer Berlin Heidelberg, Berlin, Heidelberg (1999). https://doi.org/10.1007/3-540-48405-1_34
21. Herlédan Le Merdy, A., Wesolowski, B.: The supersingular endomorphism ring problem given one endomorphism. IACR Communications in Cryptology **2**(1) (2025). https://doi.org/10.62056/akgyivrzn
22. Hofheinz, D., Hövelmanns, K., Kiltz, E.: A modular analysis of the Fujisaki-Okamoto transformation. In: Theory of Cryptography Conference. pp. 341–371. Springer (2017). https://doi.org/10.1007/978-3-319-70500-2_12
23. Jao, D., Azarderakhsh, R., Campagna, M., Costello, C., Feo, L.D., Hess, B., Jalali, A., Koziel, B., LaMacchia, B., Longa, P., Naehrig, M., Renes, J., Soukharev, V., Urbanik, D., Pereira, G., Karabina, K., Hutchinson, A.: Sike. Tech. rep., National Institute of Standards and Technology (2020), https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-3-submissions
24. Jao, D., De Feo, L.: Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. In: Yang, B.Y. (ed.) Post-Quantum Cryptography. pp. 19–34. Springer Berlin Heidelberg, Berlin, Heidelberg (2011). https://doi.org/10.1007/978-3-642-25405-5_2
25. Kani, E.: The number of curves of genus two with elliptic differentials. Journal für die reine und angewandte Mathematik **485**, 93–122 (1997), http://eudml.org/doc/183547
26. Kiltz, E., Malone-Lee, J.: A general construction of IND-CCA2 secure public key encryption. In: IMA International Conference on Cryptography and Coding. pp. 152–166. Springer (2003). https://doi.org/10.1007/978-3-540-40974-8_13
27. Kim, W., Lee, J., Kim, H., Lee, C.: Sqisign with fixed-precision integer arithmetic. Cryptology ePrint Archive (2025)
28. Kirschmer, M., Voight, J.: Algorithmic enumeration of ideal classes for quaternion orders. SIAM Journal on Computing **39**(5), 1714–1747 (2010). https://doi.org/10.1137/080734467

29. Maino, L., Martindale, C., Panny, L., Pope, G., Wesolowski, B.: A direct key recovery attack on SIDH. In: Hazay, C., Stam, M. (eds.) Advances in Cryptology – EUROCRYPT 2023. pp. 448–471. Springer Nature Switzerland, Cham (2023). https://doi.org/10.1007/978-3-031-30589-4_16

30. Moriya, T.: Is-cube: An isogeny-based compact kem using a boxed sidh diagram. Cryptology ePrint Archive (2023)

31. Moriya, T.: On weak keys of POKÉ. Cryptology ePrint Archive, Paper 2026/1002 (2026), https://eprint.iacr.org/2026/1002

32. Moriya, T., Stopar, M.: Lit-sigamal: An efficient isogeny-based pke based on a lit diagram. In: Hanaoka, G., Yang, B.Y. (eds.) Advances in Cryptology – ASIACRYPT 2025. pp. 275–306. Springer Nature Singapore, Singapore (2026). https://doi.org/10.1007/978-981-95-5113-2_9

33. Mumford, D.: Abelian varieties, tata institute of fundamental research studies in mathematics, no. 5, tata institute of fundamental research, bombay, 1970. MR **282985**(44), 219

34. Nakagawa, K., Onuki, H.: QFESTA: Efficient algorithms and parameters for FESTA using quaternion algebras. In: Reyzin, L., Stebila, D. (eds.) Advances in Cryptology – CRYPTO 2024. pp. 75–106. Springer Nature Switzerland, Cham (2024). https://doi.org/10.1007/978-3-031-68388-6_4

35. Page, A., Wesolowski, B.: The supersingular endomorphism ring and one endomorphism problems are equivalent. In: Joye, M., Leander, G. (eds.) Advances in Cryptology – EUROCRYPT 2024. pp. 388–417. Springer Nature Switzerland, Cham (2024)

36. Pope, G., Reijnders, K., Robert, D., Sferlazza, A., Smith, B.: Simpler and Faster Pairings from the Montgomery Ladder. IACR Communications in Cryptology **2025**(2) (Jul 2025). https://doi.org/10.62056/ah2i893y6, https://inria.hal.science/hal-05142445

37. Robert, D.: Breaking SIDH in polynomial time. In: Hazay, C., Stam, M. (eds.) Advances in Cryptology – EUROCRYPT 2023. pp. 472–503. Springer Nature Switzerland, Cham (2023). https://doi.org/10.1007/978-3-031-30589-4_17

38. Robert, D.: Fast pairings via biextensions and cubical arithmetic (Dec 2024), https://hal.science/hal-04848028, working paper or preprint

39. Silverman, J.H.: The arithmetic of elliptic curves, vol. 106. Springer (2009). https://doi.org/10.1007/978-0-387-09494-6

40. Voight, J.: Quaternion algebras. Springer (2021)

41. Wesolowski, B.: The supersingular isogeny path and endomorphism ring problems are equivalent. In: 2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS). pp. 1100–1111 (2022). https://doi.org/10.1109/FOCS52979.2021.00109

A Algorithms for the reduction proof

Algorithm 9: Algorithm ExtractSend

Input: A C-INKE instance x, and an efficient representation $(s_\varphi, \mathcal{E})$ of an isogeny $\varphi : E_{A_1} \to E_{A_1B}$, where E_{A_1} and E_{A_1B} are from the instance x.

Output: The sender's secret isogeny $\psi : E_0 \to E_B$ of the instance x.

Parse x as

$$E_0, (P_0, Q_0), (R_0, S_0), E_{A_1}, (R_{A_1}, S_{A_1}), E_A, (R_A, S_A), (P_A, Q_A),$$
$$E_B, (P_B, Q_B), E_{AB}, (P_{AB}, Q_{AB}).$$

Set $z \leftarrow \varphi$.
repeat
  Set $\varphi \leftarrow z$.
  Compute $z \leftarrow \mathsf{DIVIDE}((s_\varphi, \mathcal{E}), 3)$. // Theorem 7
until $z = \perp$
Compute $G = \ker \widehat{\varphi} \cap E_{A_1B}[3^b]$ by running $\mathcal{E}.\mathsf{eval}(s_\varphi, E[3^b])$.
if *G is not a cyclic subgroup of order 3^b* **then aborts**.
Compute a kernel representation of the isogeny $\widehat{\psi'} : E_{A_1B} \to E' = E_{A_1B}/G$.
Compute $\delta \in \mathbb{Z}_{3^b}^\times$ such that $\ker \psi' = \langle R_{A_1} + [\delta] S_{A_1} \rangle$
compute $\psi'' : E_A \to E''$ such that $\ker \psi'' = \langle R_A + [\delta] S_A \rangle$.
if *$j(E') \neq j(E_{A_1})$ of $j(E'') \neq j(E_{AB})$* **then aborts**.
if *$\langle \psi'(P_0) \rangle \neq \langle P_B \rangle$ or $\langle \psi'(Q_0) \rangle \neq \langle Q_B \rangle$ or $\langle \psi''(P_A) \rangle \neq \langle P_{AB} \rangle$ or $\langle \psi'(Q_B) \rangle \neq \langle Q_{AB} \rangle$* **then aborts**.
Compute $\psi : E_0 \to E_B$ such that $\ker \psi = \langle R_0 + [\delta] S_0 \rangle$.
return $\psi : E_0 \to E_B$.

Theorem 7 (Theorem 3 in [21]). *There is an algorithm* DIVIDE *which takes as input*

- *Two supersingular elliptic curves E_1, E_2 defined over $\mathbb{F}_{p^2}$,*
- *an isogeny $\varphi : E_1 \to E_2$ in efficient representation,*
- *an integer $N < \deg \varphi$,*

and returns an efficient representation of φ/N if this quotient is an isogeny, and $\perp$ otherwise. The algorithm runs in time polynomial in $k \log p$ and $\log \deg \varphi$.

Algorithm 10: Algorithm ExtractRecv

Input: A Recv instance x, and an efficient representation $(s_\varphi, \mathcal{E})$ of an isogeny $\varphi : E_{AB} \to E_{A_1B}$, where E_{AB} and E_{A_1B} are from the instance x.

Output: The receiver's secret isogeny $\phi : E_0 \to E_A$ of the instance x.

Subroutine: A factorization oracle $\mathcal{O}_{\mathsf{fact}}$.

Parse x as

$$(E_0, (P_0, Q_0), (R_0, S_0), E_{A_1}, (R_{A_1}, S_{A_1}), E_A, (P_A, Q_A), (R_A, S_A)).$$

Compute the factorization $\{(q_i, e_i)\}_{1 \le i \le \ell} \leftarrow \mathcal{O}_{\mathsf{fact}}(\deg \varphi)$ such that $\deg \varphi = \prod_{i=1}^{\ell} q_i^{e_i}$.

Let $\mathcal{D} \leftarrow \{d \mid d = \prod_{i=1}^{\ell} q_i^{f_i},\ d < 2^a \text{ and } \gcd(d, 6) = 1\}$.

for $d \in \mathcal{D}$ **do**

  Compute x such that $e(R_0, S_0)^x = e(R_{A_1}, S_{A_1})$.

  if $(x \cdot d^{-1}) \mod 3^b$ *is not quadratic* **then continue**

  Compute γ_3' such that $\gamma_3'^2 \equiv (x \cdot d^{-1}) \mod 3^b$.

  Attempt the torsion point attack using // Appendix D

$$(E_0, (R_0, S_0), E_{A_1}, ([\gamma_3'^{-1}]R_{A_1}, [\gamma_3'^{-1}]S_{A_1})).$$

  if *the attack succeeds* **then**

    recover the secret isogeny $\phi_1 : E_0 \to E_{A_1}$ of degree d.

  else

    continue

  Compute x such that $e(R_{A_1}, S_{A_1})^x = e(R_A, S_A)$.

  if $(x \cdot (2^a - d)^{-1}) \mod 3^b$ *is not quadratic* **then continue**

  Compute γ_3 such that $\gamma_3^2 \equiv (x \cdot (2^a - d)^{-1}) \mod 3^b$.

  Attempt the torsion point attack using // Appendix D

$$(E_{A_1}, (R_{A_1}, S_{A_1}), E_{A_1}, ([\gamma_3^{-1}]R_A, [\gamma_3^{-1}]S_A)).$$

  if *the attack succeeds* **then**

    recover the secret isogeny $\phi_2 : E_{A_1} \to E_A$ of degree $2^a - d$.

    return $\phi = \phi_2 \circ \phi_1$.

  else

    continue

return $\perp$.

Algorithm 11: Algorithm ExtendRecv

Input: A Recv instance x.

Output: Kernel representations of the three cyclic 3^b-isogenies $\psi : E_0 \to E_B$, $\psi' : E_{A_1} \to E_{A_1 B}$, $\psi'' : E_A \to E_{AB}$, and an associated C-INKE instance x.

1 Parse x and obtain

$$E_0, (P_0, Q_0), (R_0, S_0), E_{A_1}, (R_{A_1}, S_{A_1}), E_A, (R_A, S_A), (P_A, Q_A).$$

2 Choose a random scalar $\delta \in \mathbb{Z}_{3^b}^{\times}$.

3 Compute 3^b-isogenies $\psi : E_0 \to E_B$, $\psi' : E_{A_1} \to E_A$ and $\psi'' : E_A \to E_{AB}$ such that $\ker \psi = \langle R_0 + [\delta]S_0 \rangle$, $\ker \psi' = \langle R_{A_1} + [\delta]S_{A_1} \rangle$ and $\ker \psi'' = \langle R_A + [\delta]S_A \rangle$.

4 Choose a random scalar $\omega_2 \in \mathbb{Z}_{2^a}^{\times}$.

5 Compute

$$P_B, Q_B = [\omega_2]\psi(P_0), [\omega_2^{-1}]\psi(Q_0),$$
$$P_{AB}, Q_{AB} = [\omega_2]\psi''(P_A), [\omega_2^{-1}]\psi''(Q_A).$$

6 Set $x' =$

$$E_0, (P_0, Q_0), (R_0, S_0), E_{A_1}, (R_{A_1}, S_{A_1}), E_A, (R_A, S_A), (P_A, Q_A), E_B, (P_B, Q_B), E_{AB}, (P_{AB}, Q_{AB}).$$

7 **return** $(\psi, \psi', \psi'', x')$.

Algorithm 12: Algorithm StrSendToCINKE

Input: A Str-Send instance x.
Output: Two C-INKE instances y_1, y_2 and associated isogenies $(\phi'_1, \phi'_2, \phi'_3, \phi'_4)$.

Parse x as
$$E_0, (P_0, Q_0), \phi, E_B, (P_B, Q_B), \phi'.$$
Compute the $\deg \phi$ $(= \deg \phi')$ and find q such that $q(2^a - q) = \deg \phi$ by solving a quadratic equation.
Set $\phi_1, \phi_2, \phi_3, \phi_4$ as isogenies that satisfy $\phi = \phi_2 \circ \phi_1 = \phi_4 \circ \phi_3$, $\deg \phi_1 = \deg \phi_4 = q$, and $\deg \phi_2 = \deg \phi_3$.
Set $\phi'_1, \phi'_2, \phi'_3, \phi'_4$ as isogenies that satisfy $\phi' = \phi'_2 \circ \phi'_1 = \phi'_4 \circ \phi'_3$, $\deg \phi'_1 = \deg \phi'_4 = q$, and $\deg \phi'_2 = \deg \phi'_3$.
Sample uniformly random scalars $\alpha_2, \beta_2 \in \mathbb{Z}_{2^a}^{\times}, \gamma, \gamma' \in \mathbb{Z}_{3^b}^{\times}$.
Compute the codomain curves E_{12,A_1} and E_{34,A_1} of the isogenies ϕ_1 and ϕ_3, respectively.
Sample a random 3^b-torsion basis $R_0, S_0 \in E_0[3^b]$ and compute
$$\begin{aligned}(R_{12,A_1}, S_{12,A_1}) &\leftarrow ([\gamma']\phi_1(R_0), [\gamma']\phi_1(S_0))\\ (R_A, S_A) &\leftarrow ([\gamma]\phi(R_0), [\gamma]\phi(S_0))\\ (P_A, Q_A) &\leftarrow ([\alpha_2]\phi(P_0), [\beta_2]\phi_1(Q_0))\\ (P_{AB}, Q_{AB}) &\leftarrow ([\alpha_2]\phi'(R_0), [\beta_2]\phi'(S_0)).\end{aligned}$$
Set y_1 as
$$(E_0, (P_0, Q_0), (R_0, S_0), E_{12,A_1}, (R_{12,A_1}, S_{12,A_1}), E_A, (P_A, Q_A), (R_A, S_A)), (E_B, (P_B, Q_B)), (E_{AB}, (P_{AB}, Q_{AB})).$$
Compute
$$(R_{34,A_1}, S_{34,A_1}) \leftarrow ([\gamma']\phi_3(R_0), [\gamma']\phi_3(S_0)).$$
Set y_2 as
$$(E_0, (P_0, Q_0), (R_0, S_0), E_{34,A_1}, (R_{34,A_1}, S_{34,A_1}), E_A, (P_A, Q_A), (R_A, S_A)), (E_B, (P_B, Q_B)), (E_{AB}, (P_{AB}, Q_{AB})).$$
return $(y_1, y_2), (\phi'_1, \phi'_2, \phi'_3, \phi'_4)$.

B The adversary $\mathcal{B}$ in Theorem 4

Algorithm 13: Algorithm RecoverKey

Input: A Recv instance x, and a Str-Send instance y.
Output: A solution of the Recv problem or the Str-Send problem
State: Two tables $\mathcal{T}_{A_1}$, $\mathcal{T}_B$ and a variable σ_{known} for solving the Recv problem.
Subroutine: An adversary $\mathcal{A}$ solving C-INKE

Sample

$$(\mathsf{expl}, \mathsf{case}) \xleftarrow{\$} \{(\mathsf{end}, -), (A_1, \mathsf{good}), (A_1, \mathsf{bad}), (A, \mathsf{good}), (A, \mathsf{bad}), (B, -), (AB, \mathsf{good}), (AB, \mathsf{bad})\}.$$

if $\mathsf{expl} \in \{\mathsf{end}, B\}$ *or* $\mathsf{case} = \mathsf{bad}$ *and* $\mathsf{expl} \neq AB$ **then**
  Parse x as $(E_0, (P_0, Q_0), (R_0, S_0), E_{A_1}, (R_{A_1}, S_{A_1}), E_A, (P_A, Q_A), (R_A, S_A))$.
  Compute $(\psi, \psi', \psi'', y') \leftarrow \mathsf{ExtendRecv}(x)$. // Algorithm 11
  Compute $z \leftarrow \mathcal{A}(y')$.
  if $z = \perp$ **then abort**.
  Parse z as $(E_{A_1 B}, \varphi_{\mathsf{expl}})$.
  if $\mathrm{domain}(\varphi_{\mathsf{expl}}) \neq E_{\mathsf{expl}}$ **then abort**
  if $\mathsf{expl} \in \{\mathsf{end}, B\}$ // Case 1 and 4
  then
    if $\mathsf{expl} = \mathsf{end}$ **then** $\sigma \leftarrow \widehat{\psi'} \circ \varphi_{\mathsf{expl}}$
    if $\mathsf{expl} = B$ **then** $\sigma \leftarrow \widehat{\psi'} \circ \varphi_{\mathsf{expl}} \circ \psi$
    Compute $\mathrm{End}(E_{A_1})$ from the path $\sigma : E_{\mathsf{end}} \to E_{A_1}$. // Theorem 5
    Recover ϕ by using $\mathrm{End}(E_{A_1})$. // Lemma 4.2
    return ϕ as a solution of the Recv problem of x.
  else if $\mathsf{case} = \mathsf{bad}$ // Case 2-2 and 3-2
  then
    if $\mathsf{expl} = A_1$ **then** Store an endomorphism $\widehat{\varphi}_{\mathsf{expl}} \circ \psi'$ into $\mathcal{T}_{A_1}$.
    else if $\mathsf{expl} = A$ **then**
      if $\sigma_{\mathsf{known}} = \perp$ **then**
        Set $\sigma_{\mathsf{known}} \leftarrow \widehat{\varphi}_{\mathsf{expl}} \circ \psi'$.
        abort
      Store an endomorphism $\widehat{\sigma}_{\mathsf{known}} \circ \widehat{\varphi}_{\mathsf{expl}} \circ \psi'$ into $\mathcal{T}_{A_1}$.
    Compute a subring $\mathcal{R}$ generated by $\mathcal{T}_{A_1}$.
    if $\mathrm{rank}(\mathcal{R}) < 4$ **then abort**
    Compute $\mathrm{End}(E_{A_1})$ from $\mathcal{R}$. // Theorem 6
    Recover ϕ by using $\mathrm{End}(E_{A_1})$. // Lemma 4.2
    return ϕ as a solution of the Recv problem of x.

// continued in Algorithm 14

Algorithm 14: Algorithm RecoverKey (continued)

Input: inherited

else if case = good *and* expl $\neq AB$ **then**
  Compute $(y_1, y_2), _ \leftarrow \mathsf{StrSendToCINKE}(y)$. // Algorithm 12
  Compute $z \leftarrow \mathcal{A}(y_1)$.
  if $z = \perp$ **then abort**.
  Parse z as $(E_{A_1B}, \varphi_{\mathsf{expl}})$.
  if domain$(\varphi_{\mathsf{expl}}) \neq E_{\mathsf{expl}}$ **then abort**.
  if expl $= A_1$ // Case 2-1
  then
    Compute $\psi \leftarrow \mathsf{ExtractSend}(\varphi_{\mathsf{expl}}, y_1)$. // Algorithm 9
    return ψ as a solution of the Str-Send problem of y.
  else if expl $= A$ // Case 3-1
  then
    if $\sigma_{\mathsf{known}} \neq \perp$ **then**
      Compute $\psi \leftarrow \mathsf{ExtractSend}(\varphi_{\mathsf{expl}} \circ \sigma_{\mathsf{known}}, y_1)$. // Algorithm 9
      return ψ as a solution of the Str-Send problem of y.
    else
      Store the path $\sigma_{\mathsf{known}} \leftarrow \widehat{\varphi}_{\mathsf{expl}} \circ \psi'$.
      abort
else if expl $= AB$ // Case 5-1 and 5-2
then
  if case = good **then**
    Compute $z \leftarrow \mathcal{A}(x)$. **if** $z = \perp$ **then abort**.
    Parse z as $(E_{A_1B}, \varphi_{\mathsf{expl}})$.
    if domain$(\varphi_{\mathsf{expl}}) \neq E_{\mathsf{expl}}$ **then abort**.
    Compute $\phi \leftarrow \mathsf{ExtractRecv}(\varphi_{\mathsf{expl}}, x)$. // Algorithm 10
    return ϕ as a solution of the Recv problem of x.
  else if case = bad **then**
    Parse y as $(E_0, (P_0, Q_0), \phi, E_B, (P_B, Q_B), \phi')$.
    Compute $(y_1, y_2), (\phi'_1, \phi'_2, \phi'_3, \phi'_4) \leftarrow \mathsf{StrSendToCINKE}(y)$.
    // Algorithm 12
    if $\mathcal{T}_B = \emptyset$ **then** Compute $z \leftarrow \mathcal{A}(y_1)$.
    else Compute $z \leftarrow \mathcal{A}(y_2)$.
    if $z = \perp$ **then abort**.
    Parse z as $(E_{A_1B}, \varphi_{\mathsf{expl}})$.
    if domain$(\varphi_{\mathsf{expl}}) \neq E_{\mathsf{expl}}$ **then abort**.
    if $\mathcal{T}_B = \emptyset$ **then**
      Compute the endomorphism $\sigma = \widehat{\phi}'_1 \circ \varphi_{\mathsf{expl}} \circ \phi'$.
      Store σ into $\mathcal{T}_B$.
      abort.
    else
      Compute the endomorphism $\sigma = \widehat{\phi}'_3 \circ \varphi_{\mathsf{expl}} \circ \phi'$.
      Store σ into $\mathcal{T}_B$.
    Compute a subring $\mathcal{R}$ generated by $\mathcal{T}_B$.
    if rank$(\mathcal{R}) < 4$ **then abort**.
    Compute End(E_B) from $\mathcal{R}$. // Theorem 6
    Compute ψ by using End(E_B). // Lemma 4.1
    return ψ as a solution of the Str-Send problem of y.

Algorithm 15: Algorithm $\mathcal{B}$

Input: An instance $x \leftarrow \mathsf{C\text{-}INKE}(1^\lambda)$ and an instance $y \leftarrow \mathsf{Str\text{-}Send}(1^\lambda)$.
Output: A solution of the Recv problem or the Str-Send problem
Subroutine: An adversary $\mathcal{A}$ solving C-INKE with probability ε and the RecoverKey algorithm

Set a table $\mathcal{T}_{A_1} \leftarrow \{\}$.
Set a table $\mathcal{T}_B \leftarrow \{\}$.
Set a variable $\sigma_{\mathsf{known}} \leftarrow \perp$.
for $i \leftarrow 1$ **to** $\lceil 1/\varepsilon \rceil$ **do**
  $z \leftarrow \mathsf{RecoverKey}(x, y, \mathcal{T}_{A_1}, \mathcal{T}_B, \sigma_{\mathsf{known}}, \mathcal{A})$. // Algorithm 13
  if $z \neq \perp$ **then return** z
return $\perp$.

C Missing Proofs

C.1 Proof of Theorem 2

Proof. Our proof follows a structure similar to that of [26, Theorem 1], which is also adopted in the security analysis of POKÉ. Suppose there exists a polynomial-time algorithm $\mathcal{A}$ that breaks the IND-CPA security of our scheme. We show how to construct a polynomial-time algorithm $\mathcal{B}$ that solves the C-INKE problem using $\mathcal{A}$.

We assume that the KDF of our scheme is modeled as a random oracle. After $\mathcal{A}$ outputs two messages m_1 and m_2 as challenge candidates, $\mathcal{B}$ responds with the ciphertext $((E_B, P_B, Q_B), (E_{AB}, P_{AB}, Q_{AB}), m_b \oplus r)$ where r is chosen uniformly at random from the message space and $b \in \{0, 1\}$ is selected randomly. If $\mathcal{A}$ does not query the random oracle at the value $j(E_{A_1 B})$, then the probability that $\mathcal{A}$ distinguishes between $m_b \oplus r$ and m_b is negligible.

If $\mathcal{A}$ queries the random oracle at the value $j(E_{A_1 B})$, then $\mathcal{B}$ can use this query to solve the C-INKE problem. If $\mathcal{A}$ makes at most q_g random-oracle queries, then the probability that $\mathcal{B}$ correctly guesses the relevant query is at least $1/q_g$. Therefore, $\mathcal{B}$ solves the C-INKE problem with probability $1/q_g$, completing the reduction. □

C.2 Proof of Theorem 3

Proof. From any efficient representation of isogenies ϕ and ϕ' output by $\mathsf{Str\text{-}Send}(1^\lambda)$, one can recover their corresponding two-dimensional representations. Here, by an *efficient representation* of an isogeny, we mean that point evaluation can be performed in polynomial time. Since $\deg \phi = q(2^a - q) < p + 1$, the degree of ϕ can be determined by evaluating ϕ on a basis $P, Q \in E_0(\mathbb{F}_{p^2})$. Given $\deg \phi$, the parameter q can be recovered by solving the quadratic equation $q(2^a - q) = \deg \phi$. This uniquely determines q, and consequently allows one to reconstruct the associated two-dimensional representation.

The adversary $\mathcal{B}$ can recover the two-dimensional representation of ϕ, thus, can compute the isogenies ϕ_1 and ϕ_2 such that $\phi = \phi_2 \circ \phi_1$, $\deg \phi_1 = q$, and $\deg \phi_2 = 2^a - q$ for some $q \in [0, 2^a)$. The adversary $\mathcal{B}$ samples uniformly random scalars $\alpha_2, \beta_2 \in \mathbb{Z}_{2^a}^{\times}$, $\gamma, \gamma' \in \mathbb{Z}_{3^b}^{\times}$. Then, $\mathcal{B}$ chooses a random 3^b-torsion basis $R_0, S_0 \in E_0[3^b]$ and computes

$$\begin{aligned}(R_{A_1}, S_{A_1}) &= ([\gamma']\phi_1(R_0), [\gamma']\phi_1(S_0))\\ (R_A, S_A) &= ([\gamma]\phi(R_0), [\gamma]\phi(S_0))\\ (P_A, Q_A) &= ([\alpha_2]\phi(P_0), [\beta_2]\phi(Q_0))\\ (P_{AB}, Q_{AB}) &= ([\alpha_2]\phi'(P_B), [\beta_2]\phi'(Q_B)).\end{aligned}$$

$\mathcal{B}$ runs $\mathcal{A}$ with an input

$$\begin{aligned}&(E_0, (P_0, Q_0), (R_0, S_0)), (E_{A_1}, (R_{A_1}, S_{A_1}))(E_A, (P_A, Q_A), (R_A, S_A)),\\ &(E_B, (P_B, Q_B)), (E_{AB}, (P_{AB}, Q_{AB})).\end{aligned}$$

The adversary $\mathcal{B}$ then returns the same output produced by $\mathcal{A}$. □

C.3 Proof of Lemma 4.1

Proof. Let $\mathcal{B}$ be an adversary for the Send problem, and suppose that it additionally knows $\mathrm{End}(E_B)$. By using the known MaxOrder-to-EndRing reduction (Algorithm 7 in [41]), $\mathcal{B}$ can compute maximal orders $\mathcal{O}_0$ and $\mathcal{O}_B$ in the quaternion algebra $B_{p,\infty}$ such that

$$\mathcal{O}_0 \cong \mathrm{End}(E_0), \qquad \mathcal{O}_B \cong \mathrm{End}(E_B),$$

and $\mathcal{B}$ obtains the isomorphisms $\pi_0 : \mathrm{End}(E_0) \to \mathcal{O}_0$ and $\pi_B : \mathrm{End}(E_B) \to \mathcal{O}_B$.

$\mathcal{B}$ computes the connecting ideal I between the maximal orders $\mathcal{O}_0$ and $\mathcal{O}_B$ using the algorithm of [28, Algorithm 3.5]. Namely, I is a left $\mathcal{O}_0$-ideal which is also a right $\mathcal{O}_B$-ideal.

Next, $\mathcal{B}$ chooses a random element $\omega \in I$ such that $\gcd(\mathrm{Nrd}(\omega), 2) = 1$. $\mathcal{B}$ computes an equivalent prime ideal $J \sim I$ such that $\mathrm{Nrd}(J) = \frac{I\bar{\omega}}{\mathrm{Nrd}(I)}$, $\mathcal{O}_L(J) = \mathcal{O}_0$, and $\mathcal{O}_R(J) = \mathcal{O}'_B \cong \mathcal{O}_B$. From the isomorphism between $\mathcal{O}_B$ and $\mathcal{O}'_B$, $\mathcal{B}$ computes the isomorphism $\pi'_B : \mathrm{End}(E_B) \to \mathcal{O}'_B$.

Claim 1. (KernelToIdeal) Given any subgroup $G \subset E_0[2^a]$, $\mathcal{B}$ can construct a left $\mathcal{O}_0$-ideal I_G such that $\ker I_G = G$.

Proof. Suppose that $\theta_1, \theta_2, \theta_3, \theta_4 \in \mathrm{End}(E_0)$ is a basis of $\mathrm{End}(E_0)$. With respect to the basis $P_0, Q_0 \in E_0[2^a]$, $\mathcal{B}$ can compute matrices $M_1, M_2, M_3, M_4 \in M_{2\times 2}(\mathbb{Z}/2^a\mathbb{Z})$ corresponding to the endomorphisms

$$\theta_1|_{E_0[2^a]}, \theta_2|_{E_0[2^a]}, \theta_3|_{E_0[2^a]}, \theta_4|_{E_0[2^a]},$$

respectively by solving discrete logarithms in 2^a-torsion subgroup. Since M_i's are linearly independent, $\mathcal{B}$ can find c_1, c_2, c_3, c_4 such that

$$\ker\left(\sum_{i=1}^{4} c_i\theta_i\right) \cap E_0[2^a] = G.$$

By computing $I_G = \left(\sum_{i=1}^{4} c_i\pi_0(\theta_i)\right) \cdot \mathcal{O}_0 + 2^a \cdot \mathcal{O}_0$, we have $\ker I_G = G$. □

Claim 2. (IdealToIsogenyEval) Given any point $P \in E_0[2^a]$, $\mathcal{B}$ can compute $\phi_J(P) \in E_B[2^a]$, where ϕ_J is the isogeny corresponding to the ideal J.

Proof. By *Claim 1*, for any subgroup $G \subset E_0[2^a]$, the adversary $\mathcal{B}$ can compute the left $\mathcal{O}_0$-ideal I_G satisfying $\ker I_G = G$. Since $\gcd(\mathrm{Nrd}(J), \mathrm{Nrd}(I_G)) = 1$, $\mathcal{B}$ can compute the push-forward ideal

$$[J]_*[I_G] = J^{-1}(J \cap I_G).$$

Using the isomorphism π'_B, $\mathcal{B}$ can then determine the subgroup

$$G' = \ker([J]_*[I_G]) \subset E_B[2^a].$$

By construction, we have

$$\phi_J(G) = G'.$$

Let $\langle P_0, Q_0\rangle = E_0[2^a]$, and define

$$G_1 = \langle P_0\rangle, \qquad G_2 = \langle Q_0\rangle, \qquad G_3 = \langle P_0 + Q_0\rangle.$$

Applying the above procedure, $\mathcal{B}$ obtains the subgroups

$$\phi_J(G_1), \qquad \phi_J(G_2), \qquad \phi_J(G_3).$$

Suppose that $\mathcal{B}$ knows points

$$R_1 = [\delta_1]\phi_J(P_0), \qquad R_2 = [\delta_2]\phi_J(Q_0), \qquad R_3 = [\delta_3]\phi_J(P_0 + Q_0),$$

for some $\delta_i \in (\mathbb{Z}/2^a\mathbb{Z})^\times$. Using bilinearity of the Weil pairing, we obtain

$$\begin{aligned} e_{2^a}(R_1, R_2) &= e_{2^a}(P_0, Q_0)^{\delta_1\delta_2 \deg\phi_J}, \\ e_{2^a}(R_2, R_3) &= e_{2^a}(P_0, Q_0)^{\delta_2\delta_3 \deg\phi_J}, \\ e_{2^a}(R_3, R_1) &= e_{2^a}(P_0, Q_0)^{\delta_3\delta_1 \deg\phi_J}. \end{aligned}$$

Since $\deg\phi_J$ is invertible modulo 2^a, these pairing relations allow $\mathcal{B}$ to recover the scalars $\delta_1, \delta_2, \delta_3$. Consequently, $\mathcal{B}$ recovers the exact images

$$\phi_J(P_0) \qquad \text{and} \qquad \phi_J(Q_0).$$

Finally, because $\{P_0, Q_0\}$ forms a basis of $E_0[2^a]$, $\mathcal{B}$ can compute $\phi_J(P)$ for every $P \in E_0[2^a]$ by solving discrete logarithms in the 2^a-torsion subgroup. □

Consider the following subgroup of $\mathrm{Hom}(E_0, E_B)$:

$$H := \left\{ \phi \in \mathrm{Hom}(E_0, E_B) \;\middle|\; \begin{array}{l} \phi(P_0) = [\delta_1]P_B, \\ \phi(Q_0) = [\delta_2]Q_B \end{array} \text{ for some } \delta_1, \delta_2 \in \mathbb{Z}/2^a\mathbb{Z} \right\}.$$

The solution ψ to the given instance x of the Send problem belongs to H.

We identify the quotient $H/2^aH$ with the reductions of elements of H modulo 2^a. Equivalently, an element $\phi \in H$ is identified with its action on $E_0[2^a]$. With respect to the bases (P_0, Q_0) of $E_0[2^a]$ and (P_B, Q_B) of $E_B[2^a]$, this action is represented by a matrix

$$M_\phi \in M_{2\times 2}(\mathbb{Z}/2^a\mathbb{Z})$$

satisfying

$$\begin{pmatrix} \phi(P_0) \\ \phi(Q_0) \end{pmatrix} = M_\phi \begin{pmatrix} P_B \\ Q_B \end{pmatrix}.$$

By the definition of H, we have

$$\phi(P_0) = [\delta_1]P_B, \qquad \phi(Q_0) = [\delta_2]Q_B$$

for some $\delta_1, \delta_2 \in \mathbb{Z}/2^a\mathbb{Z}$. Hence

$$M_\phi = \begin{pmatrix} \delta_1 & 0 \\ 0 & \delta_2 \end{pmatrix}.$$

Therefore, the image of $H/2^aH$ inside $M_{2\times 2}(\mathbb{Z}/2^a\mathbb{Z})$ is contained in the submodule of diagonal matrices, which is generated by two elements. Consequently, $H/2^aH$ is generated by two isogenies modulo 2^a.

For an element $\gamma \in J \subset \mathcal{O}'_B$, the adversary $\mathcal{B}$ computes the endomorphism

$$\pi_B'^{-1}(\bar{\gamma}) \in \mathrm{End}(E_B).$$

Then, for any $P \in E_0[2^a]$, $\mathcal{B}$ can compute

$$\pi_B'^{-1}(\bar{\gamma}) \circ \phi_J(P)$$

using *Claim 2*. For $\gamma \in J$, define the isogeny

$$\phi_\gamma := [\mathrm{Nrd}(J)]^{-1} \circ \pi_B'^{-1}(\bar{\gamma}) \circ \phi_J.$$

Equivalently,

$$\ker \phi_\gamma = \ker\left(\frac{J\bar{\gamma}}{\mathrm{Nrd}(J)} \right).$$

Since $\mathcal{B}$ can evaluate $\phi_\gamma(P)$ for every $\gamma \in J$ and $P \in E_0[2^a]$, it can search for two elements $\gamma_1, \gamma_2 \in J$ such that

$$\phi_{\gamma_1}, \phi_{\gamma_2} \in H$$

and such that their reductions generate $H/2^aH$. This can be checked by evaluating $\phi_{\gamma_i}(P_0)$ and $\phi_{\gamma_i}(Q_0)$.

Define the $\mathbb{Z}$-sublattice

$$J' := \mathbb{Z}\gamma_1 + \mathbb{Z}\gamma_2 + 2^a J \subset J.$$

Then

$$J'/2^a J = (\mathbb{Z}/2^a\mathbb{Z})\gamma_1 + (\mathbb{Z}/2^a\mathbb{Z})\gamma_2.$$

In particular, $|J'/2^a J| \leq 2^{2a}$. Since J has rank 4 as a $\mathbb{Z}$-lattice, we have $|J/2^a J| = 2^{4a}$, and hence

$$[J : J'] = \frac{|J/2^a J|}{|J'/2^a J|} \geq \frac{2^{4a}}{2^{2a}} = 2^{2a}.$$

Therefore,

$$\det(J') \geq 2^{2a} \det(J).$$

The adversary $\mathcal{B}$ now searches for an element $\alpha \in J'$ satisfying

$$\mathrm{Nrd}(\alpha) = 3^b \mathrm{Nrd}(J).$$

Such an element corresponds to the sender's secret isogeny. Thus, its existence is guaranteed because the input instance given to $\mathcal{B}$ is generated by $\mathsf{C\text{-}INKE}(1^\lambda)$.

It is known that

$$\det(\mathcal{O}_0) = \frac{p}{4}, \qquad \det(J) = \det(\mathcal{O}_0)\mathrm{Nrd}(J)^2$$

see, e.g., [27, Lemmas 3 and 15]. Hence

$$\det(J') \geq 2^{2a} \cdot \frac{p}{4} \cdot \mathrm{Nrd}(J)^2.$$

By the Gaussian heuristic, the expected number of elements in J' of reduced norm at most $3^b\mathrm{Nrd}(J)$ is approximately

$$\frac{\mathrm{vol}(B(\sqrt{3^b\mathrm{Nrd}(J)}))}{\det(J')} = \frac{\pi^2 \left(\sqrt{3^b\mathrm{Nrd}(J)}\right)^4}{2\det(J')} = \frac{\pi^2 3^{2b}\mathrm{Nrd}(J)^2}{2\det(J')},$$

where $B(r)$ denotes a 4-dimensional ball of radius r. Using the lower bound on $\det(J')$, this is at most

$$\frac{\pi^2 3^{2b}\mathrm{Nrd}(J)^2}{2^{2a+1} \cdot (p/4) \cdot \mathrm{Nrd}(J)^2}.$$

A justification for the Gaussian heuristic assumption is given in Section C.7.

With the parameter choice

$$p \approx 2^{4\lambda}, \qquad 2^a \approx 2^{2\lambda}, \qquad 3^b \approx 2^{2\lambda},$$

the above bound becomes approximately

$$\frac{\pi^2 2^{4\lambda}}{2 \cdot 2^{4\lambda} \cdot (2^{4\lambda}/4)} = \frac{2\pi^2}{2^{4\lambda}}.$$

Therefore, there is at most one element $\alpha \in J'$ of reduced norm $3^b\mathrm{Nrd}(J)$, up to multiplication by ± 1. Consequently, the enumeration can be performed in expected constant time.

Finally, the element α determines a 3^b-isogeny

$$\psi_\alpha : E_0 \to E_B$$

corresponding to the equivalent ideal

$$\frac{J\bar{\alpha}}{\mathrm{Nrd}(J)},$$

whose kernel is

$$\ker \psi_\alpha = \ker\left(\frac{J\bar{\alpha}}{\mathrm{Nrd}(J)}\right).$$

Since

$$\alpha \in J' = \mathbb{Z}\gamma_1 + \mathbb{Z}\gamma_2 + 2^a J,$$

the induced isogeny satisfies

$$\psi_\alpha(\langle P_0 \rangle) = \langle P_B \rangle, \qquad \psi_\alpha(\langle Q_0 \rangle) = \langle Q_B \rangle.$$

Since ψ_α satisfies the required conditions of the sender's secret isogeny, it is the solution of the given Send instance. This completes the proof of Lemma 4.1. □

C.4 Proof of Lemma 4.2

Proof. Let $\mathcal{B}$ be an adversary fo the Recv problem, and it knows $\mathrm{End}(E_{A_1})$. By using the known MaxOrder-to-EndRing reduction (Algorithm 7 in [41]), $\mathcal{B}$ computes maximal orders $\mathcal{O}_0$ and $\mathcal{O}_B$ in the quaternion algebra $B_{p,\infty}$ such that

$$\mathcal{O}_0 \cong \mathrm{End}(E_0), \qquad \mathcal{O}_B \cong \mathrm{End}(E_B).$$

The adversary $\mathcal{B}$ first computes a connecting ideal I between the two endomorphism rings $\mathcal{O}_0 \cong \mathrm{End}(E_0)$ and $\mathcal{O}_{A_1} \cong \mathrm{End}(E_{A_1})$ using the algorithm of [28, Algorithm 3.5]. Suppose that J is an ideal equivalent to I such that $\mathrm{Nrd}(J)$ is coprime with 3.

Consider the following subgroup of $\mathrm{Hom}(E_0, E_{A_1})$:

$$H := \left\{ \phi \in \mathrm{Hom}(E_0, E_{A_1}) \;\middle|\; \begin{array}{l} \phi(R_0) = [\delta]R_{A_1}, \text{ for some } \delta \in \mathbb{Z}/3^a\mathbb{Z} \\ \phi(S_0) = [\delta]S_{A_1} \end{array} \right\}.$$

The solution ϕ_1 to the given instance x of the Recv problem belongs to H. By the similar reason in Section C.3, $H/3^bH$ is generated by one isogeny modulo 3^b. By using the similar approach of Section C.3, $\mathcal{B}$ can find an element $\gamma \in J$ such that $\phi_\gamma \in H$ and ϕ_γ generates $H/3^bH$.

Define the $\mathbb{Z}$-sublattice

$$J' := \mathbb{Z}\gamma + 3^b J \subset J.$$

Then, we have $\det(J') \geq 3^{3b} \det(J)$ by following the same process of Section C.3.

Next, $\mathcal{B}$ searches for an element $\alpha \in J'$ satisfying

$$\mathrm{Nrd}(\alpha) \leq 2^a \cdot \mathrm{Nrd}(J).$$

One of such elements corresponds to the receiver's secret isogeny. Thus, its existence is guaranteed because the input instance given to $\mathcal{B}$ is generated by $\mathsf{Recv}(1^\lambda)$.

It is known that (see [27, Lemma 3 and Lemma 15])

$$\det(\mathcal{O}_0) = \frac{p}{4}, \quad \det(J) = \det(\mathcal{O}_0)\mathrm{Nrd}(J)^2.$$

By the Gaussian heuristic, the expected number of elements in J' of reduced norm at most $2^a\mathrm{Nrd}(J)$ is approximately

$$\frac{\mathrm{vol}(B(\sqrt{2^a\mathrm{Nrd}(J)}))}{\det(J')} \leq \frac{\pi^2 \left(\sqrt{2^a\mathrm{Nrd}(J)}\right)^4}{2 \cdot 3^{3b} \cdot \det(J)} = \frac{\pi^2 2^{2a}\mathrm{Nrd}(J)^2}{2 \cdot 3^{3b} \cdot (p/4) \cdot \mathrm{Nrd}(J)^2},$$

where $B(r)$ denotes the four-dimensional ball of radius r. A justification for this heuristic assumption is given in Section C.7. Using the parameter choice

$$p \approx 2^{4\lambda}, \quad a \approx 2\lambda, \quad 3^b \approx 2^{2\lambda},$$

the above quantity is approximately

$$\frac{\pi^2 2^{4\lambda}}{2 \cdot 2^{6\lambda} \cdot (2^{4\lambda}/4)} = \frac{2\pi^2}{2^{6\lambda}}.$$

Hence, the corresponding ideal is expected to be uniquely determined up to multiplication by ± 1 in most cases. In the rare case where multiple candidates exist, $\mathcal{B}$ can use the given torsion information $[\gamma]\phi_1(R_0)$ and $[\gamma]\phi_1(S_0)$ to verify the correctness of the candidate equivalent ideal

$$J = \frac{I \cdot \overline{\alpha}}{\mathrm{Nrd}(I)}.$$

From the recovered degree

$$q = \mathrm{Nrd}(J) = \frac{\mathrm{Nrd}(\alpha)}{\mathrm{Nrd}(I)}$$

the adversary $\mathcal{B}$ recovers the masking values $\gamma, \gamma' \in \mathbb{Z}_{3^b}^\times$ satisfying

$$(R_{A_1}, S_{A_1}) = ([\gamma']\phi_1(R_0), [\gamma']\phi_1(S_0))$$

and

$$(R_A, S_A) = ([\gamma]\phi(R_0), [\gamma]\phi(S_0))$$

in the Recv problem.

Finally, by applying the torsion-point attack described in Section D to the input

$$(E_0, (R_0, S_0), E_{A_1}, (\phi_1(R_0), \phi_1(S_0))),$$

the adversary $\mathcal{B}$ recovers the isogeny ϕ_1. Similarly, by applying the same attack to

$$(E_{A_1}, (R_{A_1}, S_{A_1}), E_A, (\phi_2(R_{A_1}), \phi_2(S_{A_1}))),$$

$\mathcal{B}$ recovers the isogeny ϕ_2.

Consequently, $\mathcal{B}$ recovers the two-dimensional isogeny

$$\Phi = \phi_2 \circ \phi_1.$$

This completes the proof of Lemma 4.2. □

C.5 Proof of Lemma 4.3

To prove Lemma 4.3, we need the following two lemmata (Lemma C.1 and C.2).

Lemma C.1. *Given a division quaternion algebra B over $\mathbb{Q}$, There is no $\mathbb{Z}$-subalgebra $A \subseteq B$ of rank 3.*

Proof. Let A be a $\mathbb{Z}$-algebra of rank 3. Then it also generates a $\mathbb{Q}$-subalgebra $A \subseteq B$ with

$$\dim_{\mathbb{Q}} A = 3.$$

If A is commutative, then A is contained in a maximal subfield of B. By Corollary 7.7.11 of [40], every maximal subfield of a quaternion algebra is quadratic, hence

$$\dim_{\mathbb{Q}} A \leq 2,$$

a contradiction. Otherwise, A is noncommutative. Since B is a division algebra, A is also a division algebra. By Corollary 7.7.11 of [40], the dimension of a finite-dimensional division algebra over its center is a square, contradicting

$$\dim_{\mathbb{Q}} A = 3.$$

Therefore, the generated $\mathbb{Z}$-algebra has rank 1, 2, or 4. □

Lemma C.2. *Let $\psi : E_{A_1} \to E_{A_1 B}$ be a cyclic isogeny of degree 3^b. Let $\varphi : E_{A_1} \to E_{A_1 B}$ be a 3-reduced isogeny of arbitrary degree, i.e.* $\mathsf{DIVIDE}(\varphi, 3) = \perp$. *Let $\sigma = \widehat{\varphi} \circ \psi \in \mathrm{End}(E_{A_1})$. If $\ker \sigma \cap E_{A_1}[3^b] = E_{A_1}[3^b]$, then $\ker \widehat{\varphi} \cap E_{A_1 B}[3^b] = \ker \widehat{\psi}$. In particular, if φ is given as an explanation for a* C-INKE *instance, then the sender's secret isogeny ψ can be extracted from φ using the* ExtractSend *algorithm.*

Proof. Let $\ker \sigma \cap E_{A_1}[3^b] = E_{A_1}[3^b]$. Then, we have

$$\ker \widehat{\varphi} \supseteq \psi(E_{A_1}[3^b]) = \ker \widehat{\psi}.$$

Since the isogeny φ is 3-reduced, $\ker \widehat{\varphi} \cap E_{A_1 B}[3^k] \neq E_{A_1 B}[3^k]$ for all $k \in \mathbb{N}$, which means that $\widehat{\varphi} \cap E_{A_1 B}[3^b]$ is cyclic and equals to $\ker \widehat{\psi}$. □

Proof of Lemma 4.3.

Proof. Since scalar multiplication does not change the rank of the subring generated by

$$\{1, \sigma_1, \sigma_2\},$$

we may replace each σ_i by its 3-reduction $\widetilde{\sigma}_i$, namely an endomorphism satisfying

$$\mathsf{DIVIDE}(\widetilde{\sigma}_i, 3) = \perp$$

and $\widetilde{\sigma}_i = \mathsf{DIVIDE}(\sigma_i, 3^\ell)$ for some integer $\ell \geq 0$. Therefore, it suffices to prove the claim under the assumption that σ_1 and σ_2 are 3-reduced.

By Lemma C.2, if

$$\mathsf{ExtractSend}(\varphi_i, y_i)$$

aborts, then the subgroup

$$G = \ker \widehat{\varphi} \cap E_{A_1 B}[3^b]$$

fails one of the checks in lines 8, 12, or 13 of Algorithm 9. Equivalently,

$$\ker \widehat{\varphi} \cap E_{A_1 B}[3^b] \neq \ker \widehat{\psi}.$$

Hence, by the contrapositive of Lemma C.2, we have

$$\ker \sigma_i \cap E_{A_1}[3^b] \neq E_{A_1}[3^b].$$

It follows that

$$E_{A_1}[3^b] \supsetneq \ker \sigma_i \cap E_{A_1}[3^b] \supseteq \ker \psi_i'.$$

Since $E_{A_1}[3^b] \cong \mathbb{Z}/3^b\mathbb{Z} \times \mathbb{Z}/3^b\mathbb{Z}$, the endomorphism σ_i can be represented by a matrix

$$M_i \in M_{2\times 2}(\mathbb{Z}/3^b\mathbb{Z}) \setminus \{\mathcal{O}\},$$

according to the canonical 3^b-torsion basis on E_{A_1}. As $\ker M_i$ contains a cyclic group $\ker \psi_i'$ of order 3^b, it follows that $\det(M_i) \equiv 0 \pmod 3$. Suppose that the $\mathbb{Z}$-algebra $\mathfrak{A}$ generated by $\{1, \sigma_1, \sigma_2\}$ has rank 2. Reducing modulo 3, the rank of $\mathfrak{A}$ is at most 2 over $\mathbb{Z}/3\mathbb{Z}$. Hence, in the induced representation on the 3-torsion subgroup, there exist $a, b, c \in \mathbb{Z}/3\mathbb{Z}$, not all zero, such that

$$aI + b\tilde{M}_1 + c\tilde{M}_2 = \mathcal{O} \in M_{2\times 2}(\mathbb{Z}_3),$$

where $\tilde{M}_i = M_i \mod 3$ and I is the identity matrix. Taking determinants and using $\det(\tilde{M}_i) = 0$, we obtain

$$c^2 \det(\tilde{M}_2) = \det(aI + b\tilde{M}_1) = a^2 + ab\,\mathrm{tr}(\tilde{M}_1) + b^2 \det(\tilde{M}_1) = a\big(a + b\,\mathrm{tr}(\tilde{M}_1)\big) = 0.$$

Thus, either $a = 0$ or $a = -b\,\mathrm{tr}(\tilde{M}_1)$.

If $a = 0$, then $c\tilde{M}_2 = b\tilde{M}_1$, and hence $\ker(\tilde{M}_2) = \ker(\tilde{M}_1)$.

If $a \neq 0$, then

$$c\tilde{M}_2 = b\big(\mathrm{tr}(\tilde{M}_1)I - M_1\big),$$

and therefore

$$\ker(\tilde{M}_2) = \ker\big(\mathrm{tr}(\tilde{M}_1)I - M_1\big).$$

Consequently, there are at most two possible kernels that $\tilde{M}_2$ can have among the four cyclic subgroups of $\mathbb{Z}_3 \times \mathbb{Z}_3$.

Since $\ker(M_1)$ and $\ker(M_2)$ are uniformly random cyclic subgroups of $\mathbb{Z}_{3^b} \times \mathbb{Z}_{3^b}$, the probability that $\ker(\tilde{M}_2)$ does not belong to these two possibilities is at least $1/2$. In this case, the $\mathbb{Z}$-algebra $\mathfrak{A}$ generated by $\{1, \sigma_1, \sigma_2\}$ has rank strictly greater than 2.

By Lemma C.1, a division quaternion algebra admits no $\mathbb{Z}$-subalgebra of rank 3. Thus, the generated subring must have full rank 4. □

C.6 Proof of Lemma 4.4

Proof. By Lemma C.1, a division quaternion algebra admits no $\mathbb{Z}$-subalgebra of rank 3. Hence, it suffices to show that the generated algebra has rank greater than 2.

Suppose, for contradiction, that its rank is at most 2. Then there exist nonzero integers $c_1, c_2, c_3 \in \mathbb{Z}$ such that

$$[c_1] \circ \widehat{\phi}'_3 \circ \varphi_{34,\mathsf{expl}} \circ \phi' = [c_2] \circ \widehat{\phi}'_1 \circ \varphi_{12,\mathsf{expl}} \circ \phi' + [c_3].$$

Rearranging the equality gives

$$[c_3] = \Big([c_1] \circ \widehat{\phi}'_3 \circ \varphi_{34,\mathsf{expl}} - [c_2] \circ \widehat{\phi}'_1 \circ \varphi_{12,\mathsf{expl}}\Big) \circ \phi'.$$

Since $\deg \phi' = q(2^a - q)$, it follows that $q(2^a - q)$ divides c_3. Writing $c_3 = q(2^a - q)d_3$, we obtain

$$[d_3] \circ \widehat{\phi}' = [c_1] \circ \widehat{\phi}'_3 \circ \varphi_{34,\mathsf{expl}} - [c_2] \circ \widehat{\phi}'_1 \circ \varphi_{12,\mathsf{expl}}. \tag{1}$$

Composing appropriately and taking duals yields

$$\big([d_3] \circ \phi'_2 + [c_2] \circ \widehat{\varphi}_{12,\mathsf{expl}}\big) \circ \phi'_1 = [c_1] \circ \widehat{\varphi}_{34,\mathsf{expl}} \circ \phi'_3.$$

Therefore,

$$\phi'_3(\ker \phi'_1) \subseteq \ker([c_1] \circ \widehat{\varphi}_{34,\mathsf{expl}}).$$

By commutativity of the diagram, $\phi'_3(\ker \phi'_1) = \ker \phi'_4$, and hence

$$\ker \phi'_4 \subseteq \ker([c_1] \circ \widehat{\varphi}_{34,\mathsf{expl}}).$$

Consequently,

$$\ker \phi' \subseteq \ker([c_1] \circ \widehat{\varphi}_{34,\mathsf{expl}} \circ \phi'_3).$$

Similarly,

$$\ker \phi' \subseteq \ker([c_2] \circ \widehat{\varphi}_{12,\mathsf{expl}} \circ \phi'_1).$$

Applying $(\widehat{\phi'})^{-1}$ to (1), we obtain

$$[d_3] = [c_1] \circ \widehat{\phi'_3} \circ \varphi_{34,\mathsf{expl}} \circ (\widehat{\phi'})^{-1} - [c_2] \circ \widehat{\phi'_1} \circ \varphi_{12,\mathsf{expl}} \circ (\widehat{\phi'})^{-1}.$$

Since $\varphi_{34,\mathsf{expl}}$ is not extractable by the $\mathsf{ExtractRecv}$ algorithm (Algorithm 10), there exists a prime t such that $t \mid q$ and $t \nmid \deg \varphi_{34,\mathsf{expl}}$. Moreover, since

$$\deg \phi'_3 = 2^a - q$$

is coprime to t, the coefficient c_1 must be divisible by t in order for

$$[c_1] \circ \widehat{\phi'_3} \circ \varphi_{34,\mathsf{expl}}$$

to be divisible by $\widehat{\phi'}$. Write

$$c_1 = t^\ell d_1$$

with $t \nmid d_1$.

Let M_1 and M_2 denote the matrices corresponding to the endomorphisms

$$\widehat{\phi'_3} \circ \varphi_{34,\mathsf{expl}} \circ (\widehat{\phi'})^{-1} \qquad \text{and} \qquad \widehat{\phi'_1} \circ \varphi_{12,\mathsf{expl}} \circ (\widehat{\phi'})^{-1},$$

restricted to $E_{AB}[t^\ell]$.

Reducing the above relation modulo t^ℓ, we obtain

$$d_3 I_2 = t^\ell d_1 M_1 - c_2 M_2,$$

or equivalently,

$$t^\ell d_1 M_1 = d_3 I_2 + c_2 M_2.$$

Since ϕ'_4 is cyclic, $t \mid \deg \phi'_4 = q$, and $t \nmid \deg \varphi_{34,\mathsf{expl}}$, the matrix $t^{\ell-1} d_1 M_1$ has rank exactly 1 on $E_{AB}[t^\ell]$. On the other hand, the kernel of

$$d_3 I_2 + c_2 M_2$$

is determined by a linear combination of the matrices I_2 and M_2, and therefore there are at most two possible kernels.

Indeed, the condition

$$\det(d_3 I_2 + c_2 M_2) = 0$$

gives the quadratic equation

$$d_3^2 + d_3 c_2 (M_2) + c_2^2 \det(M_2) = 0.$$

Hence, there are at most two projective solutions

$$(d_3 : c_2) \in \mathbb{P}^1(\mathbb{Z}/t^\ell\mathbb{Z}),$$

and consequently at most two possible kernels.

On the other hand, the kernel of $t^{\ell-1} d_1 M_1$ corresponds to $\ker \phi'_4$, which is heuristically distributed uniformly among the cyclic subgroups of $E_{A_1 B}[t]$. Therefore, the probability that such a relation exists is at most

$$\frac{2}{t+1}.$$

Since $t \geq 5$, this probability is at most $\frac{2}{6}$. Consequently, the probability that the generated subring has rank 4 is at least $2/3$. □

C.7 Justification of the Gaussian Heuristic in Lemmata 4.1 and 4.2

In the proofs of Lemmata 4.1 and 4.2, we use the Gaussian heuristic to argue that a random homomorphism space $\mathrm{Hom}(E_0, E)$ does not contain unexpectedly small-degree isogenies except with negligible probability. In this section, we justify why this heuristic is reasonable in our setting. Our heuristic is motivated by the mixing behavior of supersingular isogeny graphs (see [11]).

Let $\varphi : E_0 \to E$ be a uniformly random cyclic ℓ-isogeny, and define

$$H_\ell := \{\mathrm{Hom}(E_0, E) : \exists\, \varphi : E_0 \to E \text{ with } \deg(\varphi) = \ell\}.$$

Since the number of cyclic subgroups of order ℓ in $(\mathbb{Z}/\ell\mathbb{Z})^2$ is given by the Dedekind psi function

$$\psi(\ell) = \ell \prod_{q|\ell} \left(1 + \frac{1}{q}\right),$$

the set H_ℓ has cardinality roughly $O(\ell)$.

Suppose that $\mathrm{Hom}(E_0, E) \in H_\ell$ additionally contains an isogeny

$$\psi : E_0 \to E$$

of degree $w < \ell$. Then

$$\mathrm{Hom}(E_0, E) \in H_\ell \cap H_w.$$

It is known that the supersingular q-isogeny graph, for a prime q, is a Ramanujan graph [11], and therefore has strong mixing properties. Heuristically, the supersingular ℓ- and w-isogeny graphs inherit similar mixing behavior. Consequently, the intersection

$$H_\ell \cap H_w$$

is expected to behave similarly to the intersection of two random subsets of the supersingular set.

Hence, for sufficiently large ℓ and w, the probability that a uniformly random element

$$\mathrm{Hom}(E_0, E) \in H_\ell$$

also belongs to H_w is approximately

$$\frac{|H_w|}{|\{\text{supersingular elliptic curves over } \mathbb{F}_{p^2}\}|} \approx \frac{12w}{p},$$

where we use the fact that the number of supersingular elliptic curves over $\mathbb{F}_{p^2}$ is approximately $p/12$.

Therefore, the probability that a random element of H_ℓ contains an isogeny of degree smaller than ℓ is bounded by

$$\sum_{w<\ell} \Pr[\mathrm{Hom}(E_0, E) \in H_w] = O\left(\frac{\ell^2}{p}\right).$$

The same reasoning also applies to elliptic curves with level structures like situations in Lemma 4.1 (Section C.3) and Lemma 4.2 (Section C.4). According to the result in [4], supersingular isogeny graphs with level structure also has the Ramanujan property. In that setting, one considers homomorphism spaces between supersingular elliptic curves equipped with level structures rather than supersingular elliptic curves. The number of such objects is approximately

$$p \times (\text{number of level structures of each curves}).$$

Consequently, in Lemma 4.1, the probability that a random homomorphism group between two elliptic curves with level structures has an isogeny of degree less than 3^b is approximately

$$\frac{3^{2b}}{p \cdot 3^{2b}} = \frac{1}{p} \approx \frac{1}{2^{4\lambda}},$$

which is again negligible. In Lemma 4.2, the probability that a random homomorphism group between two elliptic curves with level structures has an isogeny of degree less than 2^a is approximately

$$\frac{2^{2a}}{p \cdot 2^{3a}} = \frac{1}{2^a \cdot p} \approx \frac{1}{2^{6\lambda}}$$

Hence, the Gaussian heuristics used in Lemma 4.1 and Lemma 4.2 are also reasonable.

D Torsion point attack on the Recv problem

In this section, we describe the way to recover the receiver's secret from the given degree $q(2^a - q)$. Given an instance of the Recv problem

$$(E_0, (P_0, Q_0), (R_0, S_0), E_{A_1}, (R_{A_1}, S_{A_1}), E_A, (P_A, Q_A), (R_A, S_A)),$$

we can recover the scalar $\gamma \in (\mathbb{Z}/3^b\mathbb{Z})^\times$ such that

$$R_{A_1}, S_{A_1} = [\gamma]\phi_1(R_0), [\gamma]\phi_1(S_0),$$

by solving the following pairing equation

$$e_{3^b}(R_{A_1}, S_{A_1}) = e_{3^b}(R_0, S_0)^{\gamma^2 \deg \phi_1}$$

with the known degree $\deg \phi_1 = q$. With the following information

$$\deg \phi_1, R_0, S_0, \phi_1(R_0), \phi_1(S_0),$$

we can use the torsion point attack method. We use the following Robert's attack method in [37].

Theorem 8 (Theorem 1.1 in [37]). *Assume that we are given a decomposition* $N_\psi - N_{\mathsf{tor}} = a_1^2 + a_2^2 + a_3^2 + a_4^2$ *as a sum of four integer squares. Let* $\psi : E_1 \to E_2$ *be a* N_ψ*-isogeny defined over a finite field* $\mathbb{F}_{p^k}$*. Assume that* $E_1[N_{\mathsf{tor}}] \subset E_1(\mathbb{F}_{p^k})$ *and that we are given a basis* (P, Q) *of* $E_1[N_{\mathsf{tor}}]$ *and the image of* ψ *on this basis.*

Then there is an algorithm $\mathsf{Eval}(E_1, E_2, P, Q, \psi(P), \psi(Q), R)$ *which returns* $\psi(R)$ *for any point* R *of* $E_1(\mathbb{F}_{p^k})$ *in* $\tilde{O}(\ell_{N_{\mathsf{tor}}}^8 \log N_{\mathsf{tor}})$ *arithmetic operations over* $\mathbb{F}_{p^k}$*, where* $\ell_{N_{\mathsf{tor}}}$ *is the largest prime divisor of* N_{tor}*.*

In Theorem 8, since $N_\psi = \deg \phi_1 = q \approx 2^\lambda$ and $N_{\mathsf{tor}} = 3^b \approx 2^{2\lambda}$, it is easy to find four integers a_1, a_2, a_3, a_4 satisfying the required condition. The Eval algorithm in Theorem 8 requires to compute one 8-dimensional N_{tor}-endomorphism Φ such that

$$\Phi : E_1^4 \times E_2^4 \to E_1^4 \times E_2^4.$$

If $N_\psi = bN_{\mathsf{tor}} + a$ and both a and b are a sum of two squares, we can do a 4-dimensional attack. The Eval algorithm in Theorem 9 requires to compute one 4-dimensional N_{tor}-endomorphism Φ such that

$$\Phi : E_1^2 \times E_2^2 \to E_1^2 \times E_2^2.$$

Theorem 9 (Theorem 4.2 in [37]). *Assume that we have* $a, b > 0$ *such that* $N_{\mathsf{tor}} = bN_\psi + a$*,* $\gcd(a, b) = 1$*, and* a, b *can be written as a sum of two squares:* $a = a_1^2 + a_2^2, b = b_1^2 + b_2^2$*. Assume that* $\gcd(b, N_\psi)$ *has its odd prime divisors congruent to 1 modulo 4, and if* $2 \mid \gcd(b, N_\psi)$ *then* $4 \nmid b$*. Let* $\psi : E_1 \to E_2$ *be a* N_ψ*-isogeny defined over a finite field* $\mathbb{F}_{p^k}$*. Assume that* $E_0[N_{\mathsf{tor}}] \subset E_1(\mathbb{F}_{p^k})$ *and that we are given a basis* (P_1, P_2) *of* $E_1[N_{\mathsf{tor}}]$ *and the image of* ψ *on this basis.*

Then, given the decomposition of a *and* b *as these sums of two squares (e.g., given their factorizations), there is an algorithm*

$$\mathsf{Eval}(E_1, E_2, P, Q, \psi(P), \psi(Q), a_1, a_2, b_1, b_2)$$

which returns $\psi(R)$ *for any point* R *of* $E_1(\mathbb{F}_{p^k})$ *in* $O(\ell_{N_{\mathsf{tor}}}^4 \log \ell_{N_{\mathsf{tor}}} \log N_{\mathsf{tor}} + \log^2 N_{\mathsf{tor}})$ *arithmetic operations in* $\mathbb{F}_{p^k}$*, where* ℓ_{tor} *is the largest prime divisor of* N_{tor}*.*

E Optimized Decapsulation Algorithms

Our optimization of decapsulations leverages the fact that the decryption rejects whenever the two-dimensional isogeny does not split, and that, once P_B and Q_B are fixed, the kernel representation of the two-dimensional isogeny is uniquely determined. Using decryption as a validation check relies on the hardness of finding a splitting isogeny from a given product of elliptic curves. Algorithm 16 shows the optimized decapsulation of INKE. Algorithm 17 and 18 show the optimized decapsulation algorithms for POKÉ and PIKE using the same method. We highlight in yellow the components that are not in INKE.

Algorithm 16: Decapsulation (INKE)

Input: $s, \mathsf{sk} = (q, \alpha_2, \beta_2)$, $\mathsf{ct} = ((E_B, P_B, Q_B), (E_{AB}, P_{AB}, Q_{AB}), \mathsf{ct}')$
Output: A shared key K

$m' \leftarrow \mathsf{Dec}(\mathsf{sk}, \mathsf{ct})$
if $m' = \perp$ **then**
  $K \leftarrow H(s||\mathsf{ct})$
  return K.
$(r', \omega_2') \leftarrow G(m'||\mathsf{pk})$
$E_B', P_B', Q_B' \leftarrow \mathsf{isogen}_{3^b}(P_0, Q_0; r')$
if $j(E_B') = j(E_B)$ *and* $P_B = [\omega_2']P_B'$ *and* $Q_B = [1/\omega_2']Q_B'$ **then**
  $K \leftarrow H(m'||\mathsf{ct})$
else
  $K \leftarrow H(s||\mathsf{ct})$
return K.

Algorithm 17: Decapsulation (POKÉ)

Input: $s, \mathsf{sk} = (q, \alpha_2, \beta_2)$,
$\mathsf{ct} = ((E_B, P_B, Q_B, X_B, Y_B), (E_{AB}, P_{AB}, Q_{AB}), \mathsf{ct}')$
Output: A shared key K

$m' \leftarrow \mathsf{Dec}(\mathsf{sk}, \mathsf{ct})$
if $m' = \perp$ **then**
  $K \leftarrow H(s||\mathsf{ct})$
  return K.
$(r', \omega_2', D_5) \leftarrow G(m'||\mathsf{pk})$
$E_B', P_B', Q_B', X_B', Y_B' \leftarrow \mathsf{isogen}_{3^b}(P_0, Q_0, X_0, Y_0; r')$
if $j(E_B') = j(E_B)$ *and* $P_B = [\omega_2']P_B'$ *and* $Q_B = [1/\omega_2']Q_B'$ *and* $[X_B, Y_B]^T = D_5[X_B', Y_B']^T$ **then**
  $K \leftarrow H(m'||\mathsf{ct})$
else
  $K \leftarrow H(s||\mathsf{ct})$
return K.

Here, G and H are hash functions, and $\mathsf{isogen}_C(P; r')$ denotes a deterministic algorithm for computing and evaluating a C-isogeny with kernel $\langle R_0 + [r']S_0 \rangle$, where $R_0, S_0 \in E_0[C]$ form a canonical basis. In the case of PIKE, we have

$$C = 3^b \cdot 5^c \approx 2^{2\lambda}.$$

In particular, the decapsulation algorithm of PIKE must verify the random scalar η_D', which is used to mask the point X_{AB}. To perform this verification, one must recompute the point X_{AB} from the point X_B, which requires an additional C-isogeny computation, which makes the decapsulation procedure of PIKE slower.

Remark 5. Our optimized decapsulation algorithm prevents an IND-CCA adversary from manipulating the ciphertext by validating the codomain curve E_B

Algorithm 18: Decapsulation (PIKE)

Input: $s, \mathsf{sk} = (q, \alpha_2, \beta_2)$, $\mathsf{ct} = ((E_B, P_B, Q_B, Y_B), (E_{AB}, P_{AB}, Q_{AB}, X_{AB}), \mathsf{ct}')$
Output: A shared key K

$m' \leftarrow \mathsf{Dec}(\mathsf{sk}, \mathsf{ct})$
if $m' = \perp$ **then**
  $K \leftarrow H(s||\mathsf{ct})$
  return K.
$(r'_1, r'_2, \omega'_2, \eta'_D, \zeta'_D) \leftarrow G(m'||\mathsf{pk})$
$E'_{AB}, X'_{AB} \leftarrow \mathsf{isogen}_C(X_B; r'_1)$
$E'_B, P'_B, Q'_B, Y'_B \leftarrow \mathsf{isogen}_C(P_0, Q_0, Y_0; r'_2)$
if $j(E'_B) = j(E_B)$ *and* $P_B = [\omega'_2]P'_B$ *and* $Q_B = [1/\omega'_2]Q'_B$ *and* $X_{AB} = [\eta'_D]X'_{AB}$ *and* $Y_B = [\zeta'_D]Y'_B$ **then**
  $K \leftarrow H(m'||\mathsf{ct})$
else
  $K \leftarrow H(s||\mathsf{ct})$
return K.

and the evaluated points P_B and Q_B. However, a given supersingular elliptic curve admits six distinct but mutually isomorphic Montgomery representations, since the j-invariant is a degree-6 rational function in the Montgomery coefficient A:

$$j(E_A) = 256\frac{(A^2-3)^3}{A^2-4}.$$

For example, the map

$$\iota : (x, y) \mapsto (-x, iy)$$

defines an isomorphism between the Montgomery curves

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

and

$$\widetilde{E} : y^2 = x^3 - Ax^2 + x.$$

Therefore, to prevent an adversary from exploiting such isomorphisms to manipulate the ciphertext, elliptic curves must be encoded canonically. In particular, during binary encoding, one should deterministically select a unique representative among the six possible Montgomery forms according to a fixed canonical ordering.