LIT-SiGamal: An efficient isogeny-based PKE based on a LIT diagram

Tomoki Moriya[1] and Miha Stopar[2,3]

[1] Information Technology R&D Center, Mistubishi Electric Corporation, Japan
Moriya.Tomoki@bp.MitsubishiElectric.co.jp
[2] Université libre de Bruxelles, Belgium
[3] Ethereum Foundation
stopar.miha@gmail.com

Abstract. We propose LIT-SiGamal, a novel isogeny-based public key encryption (PKE) scheme that combines the structure of SiGamal with the recently introduced LIT diagram framework. While SiGamal relies on a commutative CSIDH-based diagram involving an auxiliary point, LIT-SiGamal replaces this with a LIT diagram – a commutative diagram consisting of large-degree horizontal isogenies and relatively low-degree vertical isogenies.
LIT-SiGamal is an efficient and secure isogeny-based PKE scheme. Our analysis suggests that it is more efficient than QFESTA, proposed by Nakagawa and Onuki. Although LIT-SiGamal appears to be less efficient than POKÉ, proposed by Basso and Maino, it offers stronger security guarantees. Moreover, we provide a Rust implementation of LIT-SiGamal, which represents the first low-level language implementation of an isogeny-based PKE scheme developed after the break of SIDH.

Keywords: isogeny-based cryptography; Kani's theorem; public key encryption;

1 Introduction

Isogeny-based cryptography is one of the leading candidates for post-quantum cryptography, relying on the hardness of the isogeny problem over supersingular elliptic curves. One of its main advantages is the potential for relatively small key sizes compared to other post-quantum cryptosystems. For instance, the isogeny-based digital signature scheme SQISign [19] is currently considered the most compact post-quantum signature scheme. However, a common drawback of isogeny-based cryptosystems is their relatively high computational cost, often resulting in slower execution times compared to other post-quantum alternatives.

SIDH [23] was a promising and efficient isogeny-based key exchange scheme. However, in 2022, a series of independent works demonstrated that SIDH could be broken in polynomial time [8,25,34]. These attacks exploit Kani's theorem [24],

which relates isogenies of elliptic curves to isogenies of abelian surfaces (i.e., dimension-2 abelian varieties).

Following the collapse of SIDH, several alternative isogeny-based schemes have been proposed. M-SIDH [21] and bin-SIDH [4] attempt to strengthen SIDH by increasing structural complexity to resist known attacks. SQISignHD [16], proposed by Dartois, Leroux, Robert, and Wesolowski, is the first scheme that uses Kani's theorem as part of the construction. Their approach provides a novel technique to represent isogenies, which shows a potential of an improvement of isogeny-based cryptosystems by higher-dimensional isogenies. FESTA [6] is the first isogeny-based public key encryption (PKE) scheme that uses Kani's theorem as a trapdoor. Based on this, QFESTA [29] improves the performance of FESTA, while IS-CUBE [26] adapts FESTA's techniques to construct a key encapsulation mechanism (KEM).

While these schemes represent important progress, they remain less efficient than SIDH. Therefore, designing more efficient isogeny-based constructions remains a compelling and essential direction for post-quantum cryptography.

1.1 Contribution

We propose a novel isogeny-based PKE scheme called LIT-SiGamal, which integrates techniques from two existing isogeny-based schemes: SiGamal [27] and IS-CUBE. SiGamal is an ElGamal-like PKE scheme constructed using a commutative diagram that incorporates an auxiliary point of smooth order. The overall structure of SiGamal can be summarized as follows:

$$\begin{array}{ccc} (E, R) & \xrightarrow{\phi_1} & (E_1, \phi_1(R)) \\ \downarrow{\scriptstyle \phi_2} & & \downarrow{\scriptstyle \phi_2'} \\ (E_2, \phi_2(R)) & \xrightarrow[\phi_1']{} & (E_3, \phi_2'(\phi_1(R))) \end{array}$$

1. Suppose that the above diagram is commutative, and that R is a point of smooth order in E.
2. Alice computes $(E, R, E_1, \phi_1(R))$ as her public key.
3. Bob computes $(E_2, \phi_2(R), E_3, \mu \cdot \phi_2'(\phi_1(R)))$ as the ciphertext, where μ is his plaintext.
4. Alice computes the isogeny $\phi_1' \colon E_2 \to E_3$ and retrieves μ by solving the discrete logarithm problem between $\mu \cdot \phi_2'(\phi_1(R))$ and $\phi_1'(\phi_2(R))$.

LIT-SiGamal leverages a LIT (Long Isogeny with Torsion) diagram as its foundational commutative structure, in place of the CSIDH diagram used in SiGamal. A LIT diagram consists of a large-degree horizontal isogeny (ϕ_1, ϕ_1') and relatively low-degree vertical isogenies (ϕ_2, ϕ_2'), as introduced in the construction of IS-CUBE. More specifically, the diagram is parameterized such that $d_1/d_2^2 \approx 2^{2\lambda}$, where d_1 denotes the degree of the horizontal isogenies, d_2 denotes the degree of the vertical isogenies, and λ is the security parameter.

Table 1. Computational time for Rust implementation of LIT-SiGamal (sec.)

λ	Public key gen.	Encryption	Decryption
128	1.33	0.03	0.09
192	4.62	0.10	0.32
256	11.28	0.24	0.78

A key advantage of LIT-SiGamal is its efficient encryption and decryption processes. From the analysis of Section 6.1, the complexities for the LIT-SiGamal encryption and decryption are $O(\lambda \log \lambda \cdot \mathbf{M}_{6\lambda})$ for the security parameter λ, where $\mathbf{M}_{6\lambda}$ is the cost of multiplication over $\mathbb{F}_{p^2}$ with a 6λ-bit prime p. From the same analysis, we show that the complexity of encryption and decryption in SIDH is $O(\lambda \log \lambda \cdot \mathbf{M}_{4\lambda})$. Therefore, we can say that LIT-SiGamal is relatively efficient.

In addition, we provide a Rust implementation of LIT-SiGamal, which is, to the best of our knowledge, the only complete low-level implementation of an isogeny-based PKE scheme in the FESTA family.

While we cannot definitively claim uniqueness, we are unaware of any other such implementation to date. Our experimental results are presented in Table 1, where we benchmark our implementation using an Apple M1 MacBook Pro clocked at 3.2 GHz. Alongside LIT-SiGamal, we also provide a collection of common isogeny-related routines written in Rust, which may be useful for other isogeny-based cryptographic constructions and experimental prototypes.

1.2 Concurrent work

Basso and Maino proposed an isogeny-based public key encryption scheme named POKÉ [5], developed in parallel with LIT-SiGamal. The two schemes share several similarities: both are based on SiGamal and incorporate auxiliary point information into a commutative diagram of isogenies. A key distinction between them lies in the treatment of the isogeny ϕ_1: POKÉ uses an isogeny of hidden degree, whereas LIT-SiGamal employs a long isogeny of explicitly known high degree.

The performance of POKÉ is significantly better than that of LIT-SiGamal. Specifically, POKÉ is estimated to be more efficient (seee Section 6.1), and it achieves smaller ciphertext and public key sizes (see Table 6). This increased efficiency stems from the fact that POKÉ can operate over a prime of size roughly 3λ bits, whereas LIT-SiGamal requires a prime of size approximately 6λ bits.

Conversely, LIT-SiGamal can be considered more secure than POKÉ. As discussed in Remark 2, the degree of ϕ_1 can also be concealed within the LIT-SiGamal framework, making this variation nearly identical to POKÉ, apart from the length of the isogeny degree. Consequently, there is no known attack that would compromise LIT-SiGamal but not POKÉ. Moreover, an adversary could potentially break POKÉ if they succeed in recovering $\deg(\phi_1)$; in contrast, LIT-SiGamal is designed to remain secure even when $\deg(\phi_1)$ is known. A more detailed discussion is provided in Section 5.3.

Organization.

We begin by introducing background material necessary for understanding the construction and analysis of LIT-SiGamal in Section 2. Section 3 reviews several isogeny-based schemes that are closely related to LIT-SiGamal. The LIT-SiGamal scheme itself is presented in Section 4, with an overview given in Section 4.1, a detailed specification in Section 4.2, the public key construction method in Section 4.3, and a compressed variant described in Section 4.

We analyze the security of LIT-SiGamal in Section 5. In particular, we define the relevant security assumptions and prove IND-CPA security in Section 5.1, and estimate appropriate prime sizes in Section 5.2. Section 5.3 discusses the relationship between LIT-SiGamal and POKÉ. Section 6 presents our Rust implementation of LIT-SiGamal, including the explicit prime parameters used (Section 6.2) and the performance results obtained from our experiments (Section B). We conclude the paper in Section 7.

2 Preliminaries

2.1 Isogenies

In this subsection, we introduce the necessary mathematical background for the construction of LIT-SiGamal, focusing in particular on the basic theory of elliptic curves. For a more detailed treatment, we refer the reader to [35].

Let p be a prime, and let k be a field of characteristic p. An *elliptic curve defined over* k is a one-dimensional abelian variety defined over k; that is, an algebraic curve equipped with a group structure that is compatible with its algebraic structure. Let n be an integer, and let $\overline{k}$ be an algebraic closure of k. The n-torsion subgroup of E is defined as

$$E[n] = \{P \in E(\overline{k}) \mid nP = 0\}.$$

If n is coprime to p, then $E[n] \cong (\mathbb{Z}/n\mathbb{Z})^2$. If $E[p] = \{0\}$, we call E a *supersingular* elliptic curve.

Let E and E' be elliptic curves defined over k. An *isogeny* $\phi\colon E \to E'$ is a surjective morphism of algebraic varieties that is also a group homomorphism, with kernel a finite subgroup of E. If ϕ is separable as a morphism of algebraic varieties, it is called a *separable isogeny*. For a separable isogeny ϕ, we have $\deg \phi = \#\ker \phi$. An *ℓ-isogeny* is a separable isogeny whose kernel is a cyclic group of order ℓ.

Let $\mathcal{E} = (E, (P_i)_i)$ and $\mathcal{E}' = (E', (P'_i)_i)$ be pairs of an elliptic curve and its ordered points. An isogeny ϕ between $\mathcal{E}$ and $\mathcal{E}'$, denoted by $\phi\colon \mathcal{E} \to \mathcal{E}'$, is an isogeny $\phi\colon E \to E'$ such that $P'_i = \phi(P_i)$ for all i. Similarly, let $\mathcal{F} = (E, (G_i)_i)$ and $\mathcal{F}' = (E', (G'_i)_i)$ be pairs of an elliptic curve and its subsets. An isogeny ϕ between $\mathcal{F}$ and $\mathcal{F}'$, denoted by $\phi\colon \mathcal{F} \to \mathcal{F}'$, is an isogeny $\phi\colon E \to E'$ such that $G'_i = \phi(G_i)$ for all i.

For any isogeny $\phi\colon E \to E'$, there exists a corresponding isogeny $\hat{\phi}\colon E' \to E$ such that $\phi \circ \hat{\phi} = [\deg \phi]$ and $\hat{\phi} \circ \phi = [\deg \phi]$, where $[n]$ denotes the multiplication-by-n map for an integer n. The isogeny $\hat{\phi}$ is called the *dual isogeny* of ϕ.

Let G be a finite subgroup of E. Then there exists a separable isogeny $\phi\colon E \to E'$ with $\ker \phi = G$. The codomain curve E' is unique up to isomorphism, and we denote a representative by E/G. Given an elliptic curve E and a finite subgroup $G \subset E$, one can explicitly compute a separable isogeny $\phi\colon E \to E/G$ with $\ker \phi = G$ [37]. Generally, from a principally polarized abelian variety A and an appropriate finite subgroup G of A, we can compute a separable isogeny $\phi\colon A \to A/G$ with $\ker \phi = G$. In particular, if $\dim A = 2$ and G is a 2-Weil subgroup isomorphic to $(\mathbb{Z}/2\mathbb{Z})^2$, we can compute a separable isogeny $\phi\colon A \to A/G$ with $\ker \phi = G$ using the formulas provided in [36] or [17].

Let the following be a commutative diagram of elliptic curves and separable isogenies:

$$\begin{array}{ccc} E & \xrightarrow{\phi_1} & E_1 \\ {\scriptstyle \phi_2}\downarrow & & \downarrow{\scriptstyle \phi_2'} \\ E_2 & \xrightarrow[\phi_1']{} & E_3 \end{array}$$

We refer to the above as an *isogeny diamond* or *SIDH diagram* if it satisfies the following conditions: $\gcd(\deg \phi_1, \deg \phi_2) = 1$, $\ker \phi_1' = \phi_2(\ker \phi_1)$, and $\ker \phi_2' = \phi_1(\ker \phi_2)$.

Proposition 1 (Kani's theorem [24]). *Let the following diagram be an isogeny diamond:*

$$\begin{array}{ccc} E & \xrightarrow{\phi_1} & E_1 \\ {\scriptstyle \phi_2}\downarrow & & \downarrow{\scriptstyle \phi_2'} \\ E_2 & \xrightarrow[\phi_1']{} & E_3 \end{array}$$

Then the isogeny $\Psi\colon E_1 \times E_2 \to E \times E_3$ defined by

$$\Psi = \begin{pmatrix} \hat{\phi}_1 & \hat{\phi}_2 \\ -\phi_2' & \phi_1' \end{pmatrix}$$

has kernel

$$\ker \Psi = \langle (\phi_1(P), \phi_2(P)), (\phi_1(Q), \phi_2(Q)) \rangle,$$

where $\{P, Q\}$ is a basis of $E[\deg \phi_1 + \deg \phi_2]$.

2.2 Isogeny problems with torsion points information

In this subsection, we introduce basic mathematical problems relevant to our work. In particular, we define the Isogeny problem and several of its variations involving torsion point information.

Definition 1 ((Supersingular) Isogeny problem)**.** Let E_1 and E_2 be supersingular elliptic curves.

The *Isogeny problem* is stated as follows:

Compute an isogeny $\phi_1 \colon E_1 \to E_2$ *from* (E_1, E_2).

Definition 2 (CSSI problem [23])**.** Let d_1 and d_2 be coprime smooth integers. Let $\mathcal{E}_i = (E_i, (P_i, Q_i))$ for $i = 1, 2$ be pairs where E_i is a supersingular elliptic curve and (P_i, Q_i) is an ordered basis of $E_i[d_2]$.

The *CSSI problem* is stated as follows:

Compute an isogeny $\phi_1 \colon \mathcal{E}_1 \to \mathcal{E}_2$ *of degree* d_1 *from* $(\mathcal{E}_1, \mathcal{E}_2, d_1, d_2)$.

The hardness of this problem underpinned the security of SIDH [23]; however, it was shown to be solvable in polynomial time when $d_2^2 \geq d_1$, due to a series of attacks on SIDH based on Kani's theorem [8,25,34]. While the general Isogeny problem is still believed to be hard, no cryptographic scheme to date relies solely on its hardness. Instead, various structured variants of the Isogeny problem have been proposed as the foundation for isogeny-based schemes. In what follows, we introduce the *CIST* and *LIT* problems.

The *CIST problem* is a variant of the Isogeny problem proposed in [6]. It asks for the computation of an isogeny between two given elliptic curves, where the images of torsion points are provided in a masked form using an appropriate matrix.

Definition 3 (CIST problem [6])**.** Let d_1 and d_2 be coprime integers. Let $\mathcal{M}$ be a sufficiently large abelian subgroup of $\mathrm{GL}_2(\mathbb{Z}/d_2\mathbb{Z})$. Let $\mathcal{E}_i = (E_1, \mathcal{B}_i)$ for $i = 1, 2$ be pairs where E_i is a supersingular elliptic curve and $\mathcal{B}_i$ is an order basis of $E_i[d_2]$. Put ${}^t\mathcal{B}'_2 := \mathbf{A} \cdot {}^t\mathcal{B}_2$ and $\mathcal{E}'_2 = (E_2, \mathcal{B}'_2)$, where $\mathbf{A}$ is a matrix in $\mathcal{M}$.

We call the following problem the *CIST problem*:

Compute an isogeny $\phi_1 \colon \mathcal{E}_1 \to \mathcal{E}_2$ *from* $(\mathcal{E}_1, \mathcal{E}'_2, d_1, d_2, \mathcal{M})$.

As candidates for the subgroup $\mathcal{M} \subset \mathrm{GL}_2(\mathbb{Z}/d_2\mathbb{Z})$, groups of diagonal matrices and circulant matrices are suggested in [6].

The *LIT problem* is another variant of the Isogeny problem. In the setting of the LIT problem, the torsion points are revealed explicitly, whereas in the CIST problem, they are masked by a matrix. Instead, the LIT problem assumes that the degree of the isogeny ϕ_1 is significantly larger than the order of the torsion subgroup.

Definition 4 (LIT Problem [26])**.** Let d_1 and d_2 be coprime integers such that $d_1 > d_2^2$. Let $\mathcal{E}_i$ be pairs of a supersingular elliptic curve and an ordered basis of its d_2-torsion subgroup.

The *LIT problem* is stated as follows:

Compute an isogeny $\phi_1 \colon \mathcal{E}_1 \to \mathcal{E}_2$ *of degree* d_1 *from* $(\mathcal{E}_1, \mathcal{E}_2, d_1, d_2)$.

According to the discussion in [26, Section 4.2], if $d_1 > d_2^2 \cdot 2^{2\lambda}$ for a given security parameter λ, then the complexity of solving the LIT problem is estimated to be $O(2^\lambda)$.

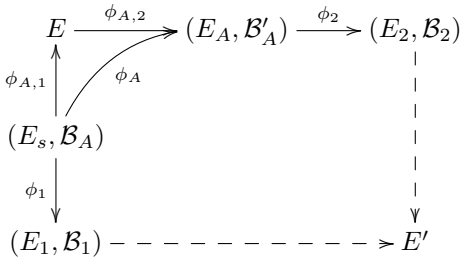
A *LIT diagram* is defined as an SIDH diagram that satisfies the degree inequality $d_1 > d_2^2 \cdot 2^{2\lambda}$.

3 Related schemes

In this section, we describe several isogeny-based schemes that are closely related to the construction of LIT-SiGamal.

3.1 FESTA

FESTA [6] is the first isogeny-based PKE scheme that uses Kani's theorem in the construction. The security of FESTA relies on the hardness of the CIST problem. Notably, they introduced the method of sending an SIDH diagram without revealing isogenies by masking torsion points by appropriate matrices. We provide a brief explanation of the outline of FESTA. The following diagram shows the basic structure of FESTA.



Assume that Bob wants to send a message to Alice using the FESTA scheme. The outline of FESTA proceeds as follows. Let ℓ_A be a small prime, and let $d_{A,1}, d_{A,2}, d_1, d_2$ be integers such that

$$d_{A,1}d_1 + d_{A,2}d_2 = \ell_A^a$$

for some $a \in \mathbb{Z}_{\geq 1}$. Let E_s be a supersingular elliptic curve, and let $\mathcal{B}_A$ be an ordered basis of $E_s[\ell_A^a]$. Finally, let $\mathcal{M}_a$ be a commutative subgroup of $\mathrm{GL}_2(\mathbb{Z}/\ell_A^a\mathbb{Z})$ of sufficiently large order.

Public key / Secret key: Alice first computes an isogeny $\phi_A \colon E_s \to E_A$ of degree $d_{A,1}d_{A,2}$ and evaluates $\phi_A(\mathcal{B}_A)$. She then selects a random matrix $\mathbf{A} \in \mathcal{M}_a$ and computes

$$^t\mathcal{B}'_A := \mathbf{A} \cdot {}^t\phi_A(\mathcal{B}_A).$$

She publishes E_A and $\mathcal{B}'_A$ as her public key. The matrix $\mathbf{A}$ serves as her secret key.

Encryption: Bob computes an isogeny $\phi_1 \colon E_s \to E_1$ of degree d_1 and an isogeny $\phi_2 \colon E_A \to E_2$ of degree d_2. He embeds his plaintext into ϕ_1 appropriately. Then, he selects a random matrix $\mathbf{B} \in \mathcal{M}_a$ and computes

$$^t\mathcal{B}_1 := \mathbf{B} \cdot {}^t\phi_1(\mathcal{B}_A), \quad {}^t\mathcal{B}_2 := \mathbf{B} \cdot {}^t\phi_2(\mathcal{B}'_A).$$

He publishes $((E_1, \mathcal{B}_1), (E_2, \mathcal{B}_2))$ as a ciphertext.

Decryption: Alice computes

$$^t\mathcal{B}'_2 := \mathbf{A}^{-1} \cdot {}^t\mathcal{B}_2.$$

By the commutativity of $\mathbf{A}$ and $\mathbf{B}$, it follows that

$$d_1\mathcal{B}'_2 = (\phi_2 \circ \phi_A \circ \hat{\phi}_1)(\mathcal{B}_1).$$

Alice then computes an (ℓ_A^a, ℓ_A^a)-isogeny Φ from $E_1 \times E_2$ with

$$\ker \Phi = \langle (d_{A,1}P_1, P'_2), (d_{A,1}Q_1, Q'_2) \rangle,$$

where we set $(P_1, Q_1) := \mathcal{B}_1$ and $(P'_2, Q'_2) := \mathcal{B}'_2$. By Kani's theorem and the identity $d_{A,1}d_1 + d_{A,2}d_2 = \ell_A^a$, it follows that

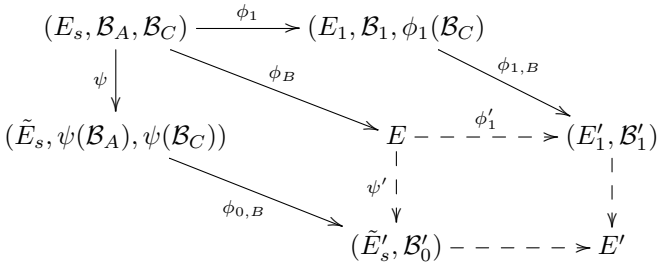
$$\Phi = \begin{pmatrix} \phi_{A,1} \circ \hat{\phi}_1 & \hat{\phi}_{A,2} \circ \hat{\phi}_2 \\ * & * \end{pmatrix},$$

where $\phi_{A,1}$ and $\phi_{A,2}$ are isogenies such that $\phi_{A,2} \circ \phi_{A,1} = \phi_A$, with $\deg \phi_{A,1} = d_{A,1}$ and $\deg \phi_{A,2} = d_{A,2}$.
Finally, Alice recovers ϕ_1 from Φ.

3.2 IS-CUBE

IS-CUBE is an isogeny-based KEM proposed in [26]. Its security relies on the hardness of both the LIT and CIST problems. A distinguishing feature of IS-CUBE is its construction of a secret SIDH diagram from random LIT diagrams and a public SIDH diagram. To transmit the secret SIDH diagram securely, IS-CUBE adopts a technique introduced in FESTA—namely, masking torsion point images using appropriately chosen matrices.

In this subsection, we present an outline of the IS-CUBE scheme. The following diagram illustrates its basic structure.



Assume that Bob wishes to establish a shared key with Alice. The IS-CUBE procedure proceeds as follows. Let ℓ_A, ℓ_B, and ℓ_C be small distinct primes, and let a, b, and c be integers such that $\ell_A^a > \ell_B^b \gg \ell_C^c$. Let $\mathcal{M}_a$ be a commutative subgroup of $\mathrm{GL}_2(\mathbb{Z}/\ell_A^a\mathbb{Z})$ of sufficiently large order.

Let E_s be a random supersingular elliptic curve, and let $\mathcal{B}_A$ be an ordered basis of $E_s[\ell_A^a]$, while $\mathcal{B}_C$ is an ordered basis of $E_s[\ell_C^c]$. Let $\psi\colon E_s \to \tilde{E}_s$ be an isogeny of degree $\ell_A^a - \ell_B^b$.

Public key / Secret key: Alice first computes a random ℓ_B^b-isogeny $\phi_1\colon E_s \to E_1$. She then computes $\phi_1(\mathcal{B}_A)$, and $\phi_1(\mathcal{B}_C)$. Next, she selects a random matrix $\mathbf{A} \in \mathcal{M}_a$ and computes

$$^t\mathcal{B}_1 := \mathbf{A} \cdot {}^t\phi_1(\mathcal{B}_A).$$

She publishes $(E_1, \mathcal{B}_1, \phi_1(\mathcal{B}_C))$ as her public key, and $\mathbf{A}$ is her secret key.

Encapsulation: Bob selects a random element $s \in (\mathbb{Z}/\ell_C^c\mathbb{Z})^\times$, and computes the following three isogenies:

$$\begin{aligned}\phi_{0,B}\colon \tilde{E}_s \to \tilde{E}'_s &:= \tilde{E}_s/\langle \psi(P_C) + s\psi(Q_C)\rangle,\\ \phi_B\colon E_s \to E &:= E_s/\langle P_C + sQ_C\rangle,\\ \phi_{1,B}\colon E_1 \to E'_1 &:= E_1/\langle \phi_1(P_C) + s\phi_1(Q_C)\rangle,\end{aligned}$$

where we set $(P_C, Q_C) := \mathcal{B}_C$. Let E be Bob's shared key. He selects a random matrix $\mathbf{B} \in \mathcal{M}_a$ and computes

$$^t\mathcal{B}'_0 := \mathbf{B} \cdot {}^t\phi_{0,B}(\psi(\mathcal{B}_A)), \quad ^t\mathcal{B}'_1 := \mathbf{B} \cdot {}^t\phi_{1,B}(\mathcal{B}_1).$$

He publishes $((\tilde{E}'_s, \mathcal{B}'_0), (E'_1, \mathcal{B}'_1))$ as a ciphertext.

Decapsulation: Alice first computes

$$^t\mathcal{B}''_1 := \mathbf{A}^{-1} \cdot {}^t\mathcal{B}'_1.$$

From the commutativity of SIDH diagrams, there exist isogenies $\psi'\colon E \to \tilde{E}'_s$ and $\phi'_1\colon E \to E'_1$ of degrees $\ell_A^a - \ell_B^b$ and ℓ_B^b, respectively, such that

$$(\ell_A^a - \ell_B^b)\mathcal{B}''_1 = (\phi'_1 \circ \hat{\psi}')(\mathcal{B}'_0).$$

Alice then computes an (ℓ_A^a, ℓ_A^a)-isogeny

$$\Phi\colon \tilde{E}'_s \times E'_1 \longrightarrow E \times *$$

with

$$\ker \Phi = \langle (P'_0, P''_1), (Q'_0, Q''_1)\rangle,$$

where we set $(P'_0, Q'_0) := \mathcal{B}'_0$ and $(P''_1, Q''_1) := \mathcal{B}''_1$. By Kani's theorem, the isogeny Φ can be written as

$$\Phi = \begin{pmatrix} \hat{\psi}' & \hat{\phi}'_1 \\ * & * \end{pmatrix}.$$

Therefore, Alice can recover the shared key E from Φ.

3.3 SiGamal

SiGamal is an isogeny-based PKE scheme proposed in [27], based on the CSIDH framework [11]. In this subsection, we provide a brief overview of the SiGamal scheme.

SiGamal is constructed by incorporating auxiliary points into a commutative isogeny diagram. The following diagram illustrates the basic structure of SiGamal. Let E_s, E_1, E_2, E_3 be elliptic curves, let L be a smooth integer, and let $R \in E_s$ be a point of order L. Let $\phi_1 \colon E_s \to E_1$, $\phi_2 \colon E_s \to E_2$, $\phi_1' \colon E_2 \to E_3$, and $\phi_2' \colon E_1 \to E_3$ be isogenies satisfying the commutativity condition $\phi_1' \circ \phi_2 = \phi_2' \circ \phi_1$. Let $\alpha, \beta \in (\mathbb{Z}/L\mathbb{Z})^\times$.

We work under the assumption that anyone who knows ϕ_1 and E_2 can compute ϕ_1'.

$$\begin{array}{ccc} (E_s, R) & \xrightarrow{\quad\phi_1\quad} & (E_1, \alpha\phi_1(R)) \\ {\scriptstyle\phi_2}\downarrow & & \downarrow{\scriptstyle\phi_2'} \\ (E_2, \beta\phi_2(R)) & \xrightarrow[\quad\phi_1'\quad]{} & (E_3, \alpha\beta\phi_2'(\phi_1(R))) \end{array}$$

Assume that Bob wishes to send a message to Alice using the SiGamal scheme. The full procedure is as follows:

Public key / Secret key: Alice first computes the top isogeny ϕ_1 and selects a random element $\alpha \in (\mathbb{Z}/L\mathbb{Z})^\times$. She then obtains (E_s, R) and $(E_1, \alpha\phi_1(R))$. She publishes $(E_s, R, E_1, \alpha\phi_1(R))$ as her public key and keeps (ϕ_1, α) as her secret key.

Encryption: Bob chooses a random element $\beta \in (\mathbb{Z}/L\mathbb{Z})^\times$ and computes the vertical isogenies ϕ_2 and ϕ_2', yielding $(E_2, \beta\phi_2(R))$ and $(E_3, \beta\phi_2'(\alpha\phi_1(R)))$. Let μ denote Bob's message, encoded as a scalar in $(\mathbb{Z}/L\mathbb{Z})^\times$. Bob computes $\mu\beta\phi_2'(\alpha\phi_1(R))$ and sends the ciphertext $(E_2, \beta\phi_2(R), E_3, \mu\beta\phi_2'(\alpha\phi_1(R)))$ to Alice.

Decryption: Using her knowledge of ϕ_1, Alice computes the isogeny ϕ_1'. She then evaluates $\phi_1'(\beta\phi_2(R))$ and multiplies the result by α to obtain

$$\alpha\phi_1'(\beta\phi_2(R)) = \beta\phi_2'(\alpha\phi_1(R)).$$

Finally, Alice recovers the message μ by solving the discrete logarithm problem between $\beta\phi_2'(\alpha\phi_1(R))$ and $\mu\beta\phi_2'(\alpha\phi_1(R))$ using the Pohlig–Hellman algorithm [32].

The above construction forms the core of the SiGamal scheme; however, it does not work in general. The issue arises because computing ϕ_1' from ϕ_1 and E_2 is, in general, hard without knowledge of ϕ_2 and ϕ_2'. In other words, it is not true in general that *anyone who knows ϕ_1 and E_2 can compute ϕ_1'*.

To address this issue, the original SiGamal scheme employs a CSIDH diagram. CSIDH is an isogeny-based key exchange scheme that relies on the action

of a specific commutative group—namely, the ideal class group—on the set of supersingular elliptic curves. The action of a group element $[\mathfrak{a}]$ on an elliptic curve E_s is computed as $E_s/E_s[\mathfrak{a}]$, where $E_s[\mathfrak{a}]$ is a finite subgroup of E_s derived from the ideal $[\mathfrak{a}]$. This structure allows us to construct a commutative diagram of isogenies by interpreting the group action as follows:

$$\begin{array}{ccc} E_s & \xrightarrow{\phi_1} & [\mathfrak{a}]E_s = E_s/E_s[\mathfrak{a}] \\ {\scriptstyle\phi_2}\downarrow & & \downarrow{\scriptstyle\phi_2'} \\ [\mathfrak{b}]E_s = E_s/E_s[\mathfrak{b}] & \overset{}{\dashrightarrow}_{\phi_1'} & [\mathfrak{a}][\mathfrak{b}]E_s \end{array}$$

The isogenies ϕ_1 and ϕ_1' correspond to the same element of the ideal class group; hence, anyone who knows ϕ_1 and E_2 can compute ϕ_1' by considering the group action on E_2 of the element related to ϕ_1.

The problem for using the CSIDH-based SiGamal is that this scheme is quite inefficient. This is because we need many high-degree isogenies to compute the group action and the size of the characteristic of the base field is large. In fact, from [14], we need more than $3,000$-bits prime for CSIDH in NIST level 1 security.

4 LIT-SiGamal

In this section, we present the construction of the LIT-SiGamal scheme.

4.1 Overview

We begin with a brief overview of the LIT-SiGamal scheme. Figure 1 summarizes its overall structure.

LIT-SiGamal is a SiGamal-based public key encryption scheme that relies on a LIT diagram, whereas the original SiGamal is constructed using a CSIDH diagram. The diagrams below illustrate the structures of the CSIDH and LIT diagrams.

A CSIDH diagram (SiGamal)

$$\begin{array}{ccc} E_s & \xrightarrow{[\mathfrak{a}]} & [\mathfrak{a}]E_s \\ {\scriptstyle[\mathfrak{b}]}\downarrow & & \downarrow{\scriptstyle[\mathfrak{b}]} \\ [\mathfrak{b}]E_s & \xrightarrow{[\mathfrak{a}]} & [\mathfrak{a}][\mathfrak{b}]E_s \end{array}$$

A LIT diagram (LIT-SiGamal)

$$\begin{array}{ccc} (E_s, \mathcal{B}_B) & \xrightarrow{\phi_1} & (E_1, \phi_1(\mathcal{B}_B)) \\ {\scriptstyle\phi_2}\downarrow & & \downarrow{\scriptstyle\phi_2'} \\ E_s/G & \overset{\text{(not computable in general)}}{\dashrightarrow}_{\phi_1'} & E_1/\phi_1(G) \end{array}$$

Public key: $(E_s, P_A, Q_A, P_B, Q_B, R), (E_1, P_1, Q_1, \phi_1(P_B), \phi_1(Q_B), \alpha\phi_1(R))$
Secret key: $(\mathbf{A}, \alpha)$
Plaintext: μ
Ciphertext: $(E'_s, P_2, Q_2, \beta\phi_2(R)), (E'_1, P_3, Q_3, \mu\beta\phi'_2(\alpha\phi_1(R)))$

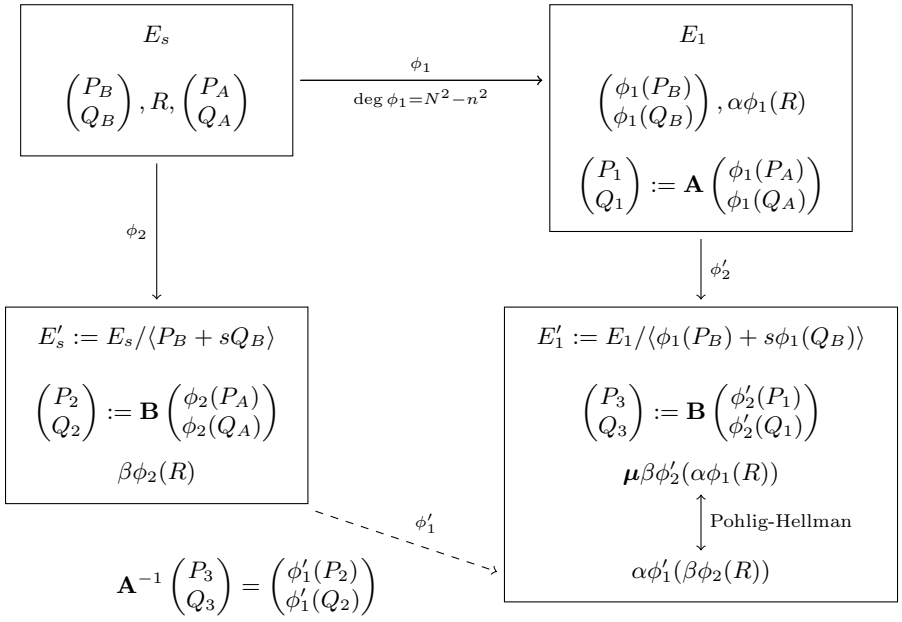


Fig. 1. Outline of LIT-SiGamal

The use of a LIT diagram instead of a CSIDH diagram offers the advantage of reducing the size of the prime p and improving the overall efficiency of the scheme. In the LIT problem setting, it is possible to compute two parallel vertical isogenies without revealing the top isogeny ϕ_1, as is required in the CSIDH setting. This suggests the potential to construct a SiGamal-based PKE scheme using a LIT diagram.

However, in general, it is not possible to compute the bottom isogeny ϕ_1' in the LIT diagram from the top isogeny ϕ_1 without knowing ϕ_2 and ϕ_2', since the diagram is not derived from a group action. This computation is required for the decryption process in SiGamal. Therefore, constructing a SiGamal-based PKE scheme from a LIT diagram is nontrivial.

To construct LIT-SiGamal, we adopt the technique used in FESTA and IS-CUBE, which involves adding auxiliary points to the diagram and masking them using a matrix. More precisely, Alice and Bob perform the following steps:

1. Alice computes a supersingular curve E_s and an isogeny $\phi_1 \colon E_s \to E_1$. She also generates an ordered basis $\mathcal{B}_B = (P_B, Q_B)$ of $E_s[M]$, where M is a smooth integer.
2. She publishes E_s, E_1, the basis $\mathcal{B}_B$, and their images $\phi_1(\mathcal{B}_B)$ under ϕ_1.
3. Let N be a sufficiently large smooth integer such that $\gcd(N, M) = 1$. Alice selects an ordered basis $\mathcal{B}_A$ of $E_s[N]$ and a matrix $\mathbf{A} \in \mathrm{GL}_2(\mathbb{Z}/N\mathbb{Z})$.
4. Alice publishes $\mathcal{B}_A$, and
$$^t\mathcal{B}_1 := \mathbf{A} \cdot {}^t\phi_1(\mathcal{B}_A),$$
along with the top part of the LIT diagram.
5. Bob selects a random $s \in (\mathbb{Z}/M\mathbb{Z})^\times$ and defines the subgroup $G = \langle P_B + sQ_B \rangle$. He computes $\phi_1(G) = \langle \phi_1(P_B) + s\phi_1(Q_B) \rangle$, and then constructs the vertical isogenies ϕ_2 and ϕ_2' with kernels G and $\phi_1(G)$, respectively.
6. Bob selects a matrix $\mathbf{B} \in \mathrm{GL}_2(\mathbb{Z}/N\mathbb{Z})$ and computes
$$^t\mathcal{B}_2 := \mathbf{B} \cdot {}^t\phi_2(\mathcal{B}_A), \quad ^t\mathcal{B}_3 := \mathbf{B} \cdot {}^t\phi_2'(\mathcal{B}_1).$$
He publishes these points along with the curves E_s/G and $E_1/\phi_1(G)$.
7. Suppose $\mathbf{AB} = \mathbf{BA}$. Then it holds that
$$^t\mathcal{B}_3 = \mathbf{A} \cdot {}^t\phi_1'(\mathcal{B}_2).$$
Alice uses $\mathbf{A}^{-1}$ to recover the image of $\mathcal{B}_2$ under ϕ_1'. Finally, using techniques from SIDH attacks, she computes the isogeny ϕ_1'.
8. Alice recovers the plaintext from ϕ_1' and image points of R.

As a result of the above procedure, Alice can compute ϕ_1' without knowledge of the vertical isogenies ϕ_2 and ϕ_2'.

In the final step of the above procedure, we apply techniques from the SIDH attacks. In general, these attacks require working with isogenies between abelian varieties of dimension 4 or 8 [34]; however, computing such high-dimensional isogenies is far less efficient than those of dimension 2.

To enable the use of isogenies of dimension 2 in the decryption process of LIT-SiGamal, we employ the following technique. Let n be a small positive integer such that $\gcd(N, n) = 1$, and let ϕ_1 be an isogeny of degree $N^2 - n^2$, where N is the order of the torsion basis $\mathcal{B}_A$. Then we obtain the following isogeny diamond:

$$\begin{array}{ccc} E_s/G & \xrightarrow{\phi_1'} & E_1/\phi_1(G) \\ {\scriptstyle [n]}\downarrow & & \downarrow{\scriptstyle [n]} \\ E_s/G & \xrightarrow{\phi_1'} & E_1/\phi_1(G) \end{array}$$

We apply the same technique introduced in [34] to the above diagram using isogenies of dimension 2. That is, we describe an isogeny Φ of dimension 2 as a composition of two isogenies whose degrees are approximately $\sqrt{\deg \Phi}$. We construct two isogenies, Ψ_0 and Ψ_1, of dimension 2, mapping from $E_s/G \times E_1/\phi_1(G)$ to an abelian variety V, with

$$\ker \Psi_0 = \langle (nP_2, P_3'), (nQ_2, Q_3') \rangle, \quad \text{and} \quad \ker \Psi_1 = \langle (nP_2, -P_3'), (nQ_2, -Q_3') \rangle,$$

where we set $(P_2, Q_2) := \mathcal{B}_2$ and define ${}^t(P_3', Q_3') = \mathbf{A}^{-1} \cdot {}^t\mathcal{B}_3$. It follows from Kani's theorem that

$$\hat{\Psi}_1 \circ \Psi_0 = \begin{pmatrix} [n] & \hat{\phi}_1' \\ -\phi_1' & [n] \end{pmatrix}.$$

Therefore, Alice can recover the isogeny ϕ_1' using the maps Ψ_0 and Ψ_1.

Remark 1. Computing the isogeny ϕ_1 is not straightforward, as $N^2 - n^2$ is generally not a smooth integer. A concrete method for constructing the public key in LIT-SiGamal is provided in Section 4.3.

4.2 Scheme of LIT-SiGamal

In this subsection, we describe the LIT-SiGamal PKE scheme. We assume that Bob wishes to send a secret message μ to Alice.

Public parameters: Let p be a prime defined by

$$p = \ell_A^a \ell_B^b \ell_C^c \cdot f - 1,$$

where ℓ_A, ℓ_B, and ℓ_C are pairwise coprime integers, and f is a small positive integer. Let $\mathcal{M}_a$ denote the subgroup of diagonal matrices in the general linear group of degree 2 over $\mathbb{Z}/\ell_A^a\mathbb{Z}$.[4] Define the plaintext space $\mathcal{P}$ as the unit group of $\mathbb{Z}/\ell_C^c\mathbb{Z}$, i.e., $\mathcal{P} = (\mathbb{Z}/\ell_C^c\mathbb{Z})^\times$.

[4] Circulant matrices can also be used in the construction of LIT-SiGamal; however, we opt for diagonal matrices to simplify both the implementation and the security analysis.

Public key / Secret key: Alice constructs her public key using the method described in Section 4.3. She begins by generating a pair of supersingular elliptic curves over $\mathbb{F}_{p^2}$, denoted by (E_s, E_1), such that there exists a cyclic isogeny $\phi_1 \colon E_s \to E_1$ of degree $\ell_A^{2a} - n^2$ for some integer n. Here, any positive integer n may be chosen, provided that $\ell_A^{2a} - n^2 > \ell_B^{2b} \cdot 2^{2\lambda}$, where λ is the security parameter.
Let $\mathcal{B}_A$ be an ordered basis of $E_s[\ell_A^a]$, and let $\mathcal{B}_B = (P_B, Q_B)$ be an ordered basis of $E_s[\ell_B^b]$. Let $R \in E_s$ be a random point of order ℓ_C^c. Alice samples a random matrix $\mathbf{A} \in \mathcal{M}_a$ and a random scalar $\alpha \in (\mathbb{Z}/\ell_C^c\mathbb{Z})^\times$. She then computes

$$ {}^t\mathcal{B}_1 := \mathbf{A} \cdot {}^t\phi_1(\mathcal{B}_A), \quad R_1 := \alpha\phi_1(R). $$

Alice publishes the tuple

$$ (E_s, \mathcal{B}_A, \mathcal{B}_B, R), \quad (E_1, \mathcal{B}_1, \phi_1(\mathcal{B}_B), R_1) $$

as her public key, and keeps $(\mathbf{A}, \alpha)$ as her secret key.

Encryption: Bob selects the plaintext μ from the plaintext space $(\mathbb{Z}/\ell_C^c\mathbb{Z})^\times$. He samples random elements $s \in (\mathbb{Z}/\ell_B^b\mathbb{Z})^\times$, $\mathbf{B} \in \mathcal{M}_a$, and $\beta \in (\mathbb{Z}/\ell_C^c\mathbb{Z})^\times$. He then computes the following two ℓ_B-isogenies:

$$ \begin{aligned} \phi_2 &\colon E_s \longrightarrow E_s' := E_s/\langle P_B + sQ_B \rangle, \\ \phi_2' &\colon E_1 \longrightarrow E_1' := E_1/\langle \phi_1(P_B) + s\phi_1(Q_B) \rangle, \end{aligned} $$

as well as the points

$$ {}^t\mathcal{B}_2 := \mathbf{B} \cdot {}^t\phi_2(\mathcal{B}_A), \quad {}^t\mathcal{B}_3 := \mathbf{B} \cdot {}^t\phi_2'(\mathcal{B}_1), \quad R' := \beta\phi_2(R), \quad R_1' := \mu\beta\phi_2'(R_1). $$

Bob sends the ciphertext

$$ (E_s', \mathcal{B}_2, R'), \quad (E_1', \mathcal{B}_3, R_1') $$

to Alice.

Decryption: Alice first computes

$$ {}^t(P_2', Q_2') := \mathbf{A} \cdot {}^t\mathcal{B}_2, \quad R'' := \alpha R'. $$

She then constructs the following two isogenies:

$$ \begin{aligned} \Psi_0 &\colon E_s' \times E_1' \longrightarrow V \quad \text{with} \quad \ker \Psi_0 = \langle (nP_2', P_3), (nQ_2', Q_3) \rangle, \\ \Psi_1 &\colon E_s' \times E_1' \longrightarrow V \quad \text{with} \quad \ker \Psi_1 = \langle (nP_2', -P_3), (nQ_2', -Q_3) \rangle, \end{aligned} $$

where we set $(P_3, Q_3) := \mathcal{B}_3$ and V is an abelian variety of dimension 2.
Let

$$ R_1'' := \mathsf{pr}_2 \circ \hat{\Psi}_1 \circ \Psi_0((-R'', 0)), $$

where pr_2 denotes the projection $E_s' \times E_1' \to E_1'$.
Finally, Alice recovers the plaintext μ' by solving the discrete logarithm problem between R_1' and R_1'', i.e., finding $\mu' \in (\mathbb{Z}/\ell_C^c\mathbb{Z})^\times$ such that

$$ R_1' = \mu' R_1''. $$

Alice outputs μ' as the decrypted message.

Table 2. Original and compressed LIT-SiGamal

	Original	Compressed
Public key	$\mathbb{F}_{p^2}^{12}$	$\mathbb{F}_{p^2}^{2} \times (\mathbb{Z}/\ell_A^a\mathbb{Z})^3 \times (\mathbb{Z}/\ell_B^b\mathbb{Z})^3 \times (\mathbb{Z}/\ell_C^c\mathbb{Z})^2 \times \{0,1\}^3$
Ciphertext	$\mathbb{F}_{p^2}^{8}$	$\mathbb{F}_{p^2}^{2} \times (\mathbb{Z}/\ell_A^a\mathbb{Z})^6 \times (\mathbb{Z}/\ell_C^c\mathbb{Z})^3 \times \{0,1\}^3$

Remark 2. In the above construction, the degree of ϕ_1 is not hidden. However, since there are many possible choices for the integer n, it is possible to hide $\deg(\phi_1)$ in the LIT-SiGamal setting. To achieve this, one must use $\gamma\phi_1(P_B)$ and $\gamma\phi_1(Q_B)$ for some $\gamma \in (\mathbb{Z}/\ell_B^b\mathbb{Z})^\times$ in place of $\phi_1(P_B)$ and $\phi_1(Q_B)$ in the public key. This modification is necessary because the Weil pairing can reveal $\deg(\phi_1)$ from the values of $\phi_1(P_B)$ and $\phi_1(Q_B)$.

Theorem 1. *LIT-SiGamal is correct.*

Proof. Let ϕ_1' be an isogeny of degree $\ell_A^{2a} - n^2$ from E_s' to E_1' satisfying $\phi_1' \circ \phi_2 = \phi_2' \circ \phi_1$. The kernel of an isogeny $\Phi\colon E_s' \times E_1' \to E_s' \times E_1'$ represented by

$$\Phi = \begin{pmatrix} [n] & \hat{\phi_1'} \\ -\phi_1' & [n] \end{pmatrix}$$

is $\langle (nP, \phi_1'(P)) \mid P \in E_s'[\ell_A^{2a}] \rangle$. Since the kernel of $\hat{\Phi}$ is

$$\Phi(\langle (nP, 0) \mid P \in E_s'[\ell_A^{2a}] \rangle) = \langle (nP, -\phi_1'(P)) \mid P \in E_s'[\ell_A^{2a}] \rangle,$$

we have $\Phi = \hat{\Psi_1} \circ \Psi_0$. Therefore, the point R_1'' is an image of $R'' = \alpha\beta\phi_2(R)$ under ϕ_1'. Since it holds that $R_1' = \mu\alpha\beta\phi_2' \circ \phi_1(R)$, it is clear that $\mu = \mu'$. □

Remark 3. To reduce the computational cost of the scheme, we often represent points in elliptic curves by their x-coordinates. In this case, we cannot distinguish R_1' and $-R_1'$ because we forget their y-coordinates. Therefore, at the end of the decryption process of LIT-SiGamal, Alice may obtain $-\mu$ instead of μ. We do not care about this error in practice since this error is easily corrected.

Remark 4. We can compress LIT-SiGamal by the same way as for SIDH [15]. That is, we define a canonical basis of an N-torsion subgroup of a given curve E and represent a basis of $E[N]$ by a matrix in $\mathrm{GL}_2(\mathbb{Z}/N\mathbb{Z})$ using the Pohlig-Hellman algorithm.

Table 2 summarizes the sets whose elements represent the public key and ciphertext in both the compressed and original versions of LIT-SiGamal.

We have implemented compressed LIT-SiGamal in SageMath. The Rust implementation of the compression is our future work.

4.3 Construction of the LIT-SiGamal public key

To construct the LIT-SiGamal public key, we must compute an isogeny of degree $\ell_A^{2a} - n^2$, which is generally not a smooth integer. To accomplish this, we can apply

techniques similar to those used in IS-CUBE, as described in [26, Sections 3.2 and 3.3]. Specifically, we compute the desired isogeny by exploiting the structure of the endomorphism ring of a curve with j-invariant 1728 defined over $\mathbb{F}_{p^2}$.

We assume that $p < (\ell_A^{2a} - n^2)\ell_B^b$. This assumption holds for our proposed parameters (see Section 5.2). Under this condition, we can compute integers x, y, z, w satisfying

$$x^2 + y^2 + p(z^2 + w^2) = (\ell_A^{2a} - n^2)\ell_B^b$$

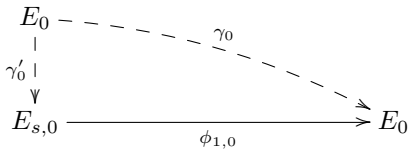
using Cornacchia's algorithm (see [19, Algorithm 1] for more details). Let E_0 be a supersingular elliptic curve with j-invariant 1728, and let π_p denote the p-Frobenius endomorphism of E_0. Let ι be an endomorphism of E_0 satisfying $\iota^2 = [-1]$. Define the endomorphism $\gamma_0 \colon E_0 \to E_0$ as

$$\gamma_0 := [x] + [y]\iota + \pi_p\big([z] + [w]\iota\big).$$

Then, we have $\deg(\gamma_0) = (\ell_A^{2a} - n^2)\ell_B^b$. Let γ_0' be a separable isogeny from E_0 with kernel $\ker(\gamma_0') = \ker(\gamma_0) \cap E_0[\ell_B^b]$, and let $E_{s,0}$ denote the codomain of γ_0'. Then there exists an isogeny $\phi_{1,0} \colon E_{s,0} \to E_0$ of degree $\ell_A^{2a} - n^2$ such that

$$\gamma_0 = \phi_{1,0} \circ \gamma_0',$$

as illustrated in the following diagram.



To construct the public key of LIT-SiGamal, we must compute the images under $\phi_{1,0}$ of points of orders ℓ_A^a, ℓ_B^b, and ℓ_C^c. Let P be a point in $E_{s,0}$ whose order is coprime to ℓ_B. In this case, we can compute $\phi_{1,0}(P)$ efficiently, since the identity

$$\gamma_0 \circ \hat{\gamma_0'} = [\ell_B^b] \circ \phi_{1,0}$$

holds. In particular, this allows us to compute the images of points of orders ℓ_A^a and ℓ_C^c directly.

If P has order divisible by ℓ_B, then $\phi_{1,0}(P)$ can still be computed using the same method employed in the decryption process. More precisely, since we can compute $\phi_{1,0}(\mathcal{B}_A)$ for a basis $\mathcal{B}_A$ of $E_{s,0}[\ell_A^a]$ and we know that $\deg(\phi_{1,0}) = \ell_A^{2a} - n^2$, the image $\phi_{1,0}(P)$ can be recovered via a two-dimensional isogeny of the form

$$\begin{pmatrix} [n] & \hat{\phi}_{1,0} \\ -\phi_{1,0} & [n] \end{pmatrix}$$

constructed from the pairs $\mathcal{B}_A$ and $\phi_{1,0}(\mathcal{B}_A)$.

Using the above method, we can construct the public key for LIT-SiGamal; however, this approach raises security concerns because it relies on the curve with j-invariant 1728.

One concern is the use of a curve with j-invariant 1728. Castryck and Vercauteren demonstrated in [13] that FESTA becomes vulnerable when such a curve is used, provided that the public points satisfy a specific condition—namely, that they are eigenvectors of a small-degree endomorphism of the curve with j-invariant 1728.

Another concern relates to the distribution of the curve $E_{s,0}$. The number of endomorphisms γ_0 with norm $(\ell_A^{2a} - n^2)\ell_B^b$—or equivalently, the number of such elements in the quaternion order—is smaller than $2^{2\lambda}$, since

$$\begin{aligned} &\#\{(x,y,z,w) \in \mathbb{Z}^4 \mid x^2 + y^2 + p(z^2 + w^2) = \ell_B^b(\ell_A^{2a} - n^2)\} \\ &\leq \#\{(z,w) \in \mathbb{Z}^2 \mid z^2 + w^2 \leq \ell_B^b(\ell_A^{2a} - n^2)/p\} \\ &< \left(2\sqrt{\frac{\ell_B^b(\ell_A^{2a} - n^2)}{p}}\right)^2 \approx 2^{2\lambda}. \end{aligned}$$

Since the number of isogenies of degree $\ell_A^{2a} - n^2$ from E_0 is about $2^{6\lambda}$, we obtain $E_{s,0}$ from a tiny subset of the set of possible elliptic curves. Therefore, it is recommended to use random elliptic curves for the public key.

We now describe the method for constructing the public key using random elliptic curves. The idea follows the approach in [26, Section 3.3] and is illustrated in the following diagram.

$$\begin{array}{ccccccccc} E_0 & \xrightarrow{\gamma_0'} & E_{s,0} & \xrightarrow{\phi_{B,1}} & E_{s,1} & \xrightarrow{\phi_{B,2}} & \cdots & \xrightarrow{\phi_{B,L}} & E_s := E_{s,L} \\ & \searrow^{\gamma_0} & \downarrow{\scriptstyle \phi_{1,0}} & & \downarrow{\scriptstyle \phi_{1,1}} & & & & \downarrow{\scriptstyle \phi_1 := \phi_{1,L}} \\ & & E_0 & \xrightarrow[\phi'_{B,1}]{} & E_{0,1} & \xrightarrow[\phi'_{B,2}]{} & \cdots & \xrightarrow[\phi'_{B,L}]{} & E_1 := E_{1,L} \end{array}$$

More precisely, we perform the following procedure:

Step 0: Compute γ_0 and γ_0' to construct $E_{s,0}$, the isogeny $\phi_{1,0}$, and the auxiliary points $\mathcal{B}_{A,0}$, $\mathcal{B}_{B,0}$, $R_{C,0}$, along with their images $\phi_{1,0}(\mathcal{B}_{A,0})$, $\phi_{1,0}(\mathcal{B}_{B,0})$, and $\phi_{1,0}(R_{C,0})$. Here, $\mathcal{B}_{A,0}$ is an ordered basis of $E_{s,0}[\ell_A^a]$, $\mathcal{B}_{B,0}$ is an ordered basis of $E_{s,0}[\ell_B^b]$, and $R_{C,0}$ is a point of order ℓ_C^c. Let $E_{0,0}$ denote E_0, a curve with j-invariant 1728.

Step $3i-2$ ($1 \leq i \leq L$): Randomly compute two ℓ_B^b-isogenies $\phi_{B,i}$ and $\phi'_{B,i}$ from $E_{s,i-1}$ and $E_{0,i-1}$, respectively, such that

$$\ker(\phi'_{B,i}) = \phi_{1,i-1}(\ker(\phi_{B,i})),$$

and ensure that this is done without backtracking. Denote the codomain of $\phi_{B,i}$ (respectively, $\phi'_{B,i}$) by $E_{s,i}$ (respectively, $E_{0,i}$). Observe that there exists an isogeny $\phi_{1,i}$ of degree $\ell_A^{2a} - n^2$ between $E_{s,i}$ and $E_{0,i}$.

Step $3i-1$ ($1 \leq i \leq L$): Compute the images of $\mathcal{B}_{A,i-1}$, and $R_{C,i-1}$ under the isogeny $\phi_{B,i}$. Define

$$\mathcal{B}_{A,i} := \phi_{B,i}(\mathcal{B}_{A,i-1}), \quad R_{C,i} := \phi_{B,i}(R_{C,i-1}).$$

Next, compute $\phi_{1,i}(\mathcal{B}_{A,i})$, and $\phi_{1,i}(R_{C,i})$ by evaluating the images of $\phi_{1,i-1}(\mathcal{B}_{A,i-1})$, and $\phi_{1,i-1}(R_{C,i-1})$ under $\phi'_{B,i}$.

Step 3*i* ($1 \leq i \leq L$): Generate a random basis $\mathcal{B}_{B,i}$ of $E_{s,i}[\ell_B^b]$, and compute the images $\phi_{1,i}(\mathcal{B}_{B,i})$ using the decryption method of LIT-SiGamal. This computation is carried out using the pairs $\mathcal{B}_{A,i}$ and $\phi_{1,i}(\mathcal{B}_{A,i})$.

Step 3L + 1: Mask the auxiliary points appropriately, and output $E_{s,L}$, $E_{0,L}$, along with the masked auxiliary points, as the public key.

Here, the integer L denotes the number of iterations.

Remark 5. It may seem that we cannot obtain two random elliptic curves using the above method, since it starts from one random elliptic curve and derives the other based on it. However, [2, Theorem 7.3] and [3, Theorem 3] show that the supersingular ℓ-isogeny graph with a level-$(\ell_A^{2a} - n^2)$ Borel structure is connected and satisfies the Ramanujan property.

Therefore, we can obtain a random supersingular elliptic curve E and a random cyclic subgroup of E of order $(\ell_A^{2a} - n^2)$ by performing a random walk on the graph. In other words, if L is sufficiently large, the above method yields a nearly uniform random pair of $(\ell_A^{2a} - n^2)$-isogenous supersingular elliptic curves.

The mixing rate of the graph is $1/\sqrt{\ell_B}$, as shown in [1, Theorem 1.1], i.e.,

$$\frac{1}{\sqrt{\ell_B}} = \limsup_{l\to\infty} \max_{u,v\in G} \left| P_{u,v}^{(l)} - \frac{1}{\#G} \right|^{\frac{1}{l}},$$

where G is the set of vertices of the graph, and $P_{u,v}^{(l)}$ is the probability that a length-l non-backtracking random walk from u reaches v.

We therefore recommend choosing L such that $(1/\sqrt{\ell_B})^{bL} \leq 1/2^\lambda$ and $\ell_B^{bL} \geq \#G \approx 2^{12\lambda}$. That is, we set $L \approx \log_{\ell_B}(2^{12\lambda})/b = 12\lambda/\log_2(\ell_B^b) \approx 6$.

5 Security analysis of LIT-SiGamal

We provide security analysis of LIT-SiGamal in this section.

5.1 Security assumptions

In this subsection, we introduce the security assumptions underlying LIT-SiGamal and prove that it achieves IND-CPA security. Specifically, we show that LIT-SiGamal is IND-CPA secure under the LIT-DDH assumption, which is analogous to the standard DDH assumption but defined in the context of a LIT diagram.

Let p be a prime of the form $p = \ell_A^a \ell_B^b \ell_C^c f - 1$, where ℓ_A, ℓ_B, and ℓ_C are distinct small primes, and f is a small integer. Let n be a small integer. Let $\mathcal{M}_a$ be the subgroup of diagonal matrices in $\mathrm{GL}_2(\mathbb{Z}/\ell_A^a\mathbb{Z})$, and let $\mathcal{M}_{a,c}$ be the subgroup of diagonal matrices in $\mathrm{GL}_2(\mathbb{Z}/\ell_A^a\ell_C^c\mathbb{Z})$.

We begin by introducing the Computational and Decisional LIT-SiGamal assumptions (Definitions 5 and 6), which serve as foundational assumptions for analyzing the security of LIT-SiGamal.

Definition 5 (Computational LIT-SiGamal assumption)**.** Let E_s and E_1 be supersingular elliptic curves with an isogeny $\phi_1 \colon E_s \to E_1$ of degree $\ell_A^{2a} - n^2$. Let $\mathcal{B}_A$ be an ordered basis of $E_s[\ell_A^a]$, let $\mathcal{B}_B = (P_B, Q_B)$ be an ordered basis of $E_s[\ell_B^b]$, and let R be a point of E_s of order ℓ_C^c. Define

$$ {}^t\mathcal{B}_{1,A} := \mathbf{A} \cdot {}^t\mathcal{B}_A, \quad \mathcal{B}_{1,B} := \phi_1(\mathcal{B}_B), \quad \text{and} \quad R_1 := \alpha\phi_1(R), $$

where $\mathbf{A}$ is a matrix in $\mathcal{M}_a$ and α is an element in $(\mathbb{Z}/\ell_C^c\mathbb{Z})^\times$. Let s be an element in $(\mathbb{Z}/\ell_B^b\mathbb{Z})^\times$, let $\phi_2 \colon E_s \to E_s'$ be a separable isogeny with $\ker\phi_2 = \langle P_B + sQ_B \rangle$, and let $\phi_2' \colon E_1 \to E_1'$ be a separable isogeny whose kernel is given by $\langle P_{1,B} + sQ_{1,B} \rangle$ where we set $(P_{1,B}, Q_{1,B}) := \mathcal{B}_{1,B}$. Define

$$ {}^t\mathcal{B}_A' := \mathbf{B} \cdot {}^t\phi_2(\mathcal{B}_A), \quad {}^t\mathcal{B}_{1,A}' := \mathbf{B} \cdot {}^t\phi_2'(\mathcal{B}_{1,A}), \quad R' := \beta\phi_2(R), \quad R_1' := \beta\phi_2'(R_1), $$

where $\mathbf{B}$ is a matrix in $\mathcal{M}_a$ and β is an element in $(\mathbb{Z}/\ell_C^c\mathbb{Z})^\times$. We denote the tuple

$$ ((E_s, \mathcal{B}_A, \mathcal{B}_B, R), (E_1, \mathcal{B}_{1,A}, \mathcal{B}_{1,B}, R_1), (E_s', \mathcal{B}_A', R'), (E_1', \mathcal{B}_{1,A}', R_1')) $$

by $T_{\text{LIT-SiGamal}}$ and denote the set of the tuples by $\mathcal{S}_{\text{LIT-SiGamal}}$. We also denote by $T_{\text{LIT-SiGamal}}(\mu)$ the tuple that is defined by replacing R_1' in $T_{\text{LIT-SiGamal}}$ with $\mu R_1'$ for $\mu \in (\mathbb{Z}/\ell_C^c\mathbb{Z})^\times$.

We call the following assumption the *Computational LIT-SiGamal assumption*:

Any probabilistic polynomial time (PPT) algorithm $\mathcal{A}$ satisfies

$$ \Pr\left[\mu = \mu^* \;\middle|\; \begin{array}{l} \mu \xleftarrow{\$} (\mathbb{Z}/\ell_C^c\mathbb{Z})^\times,\ T_{\text{LIT-SiGamal}} \xleftarrow{\$} \mathcal{S}_{\text{LIT-SiGamal}}, \\ \mu^* \leftarrow \mathcal{A}\left(T_{\text{LIT-SiGamal}}(\mu)\right) \end{array} \right] < \text{negl}(\lambda), $$

where the notation $X \xleftarrow{\$} Y$ means that X is sampled uniformly at random from Y and $\text{negl}(\cdot)$ *is a negligible function.*

We have the following proposition immediately from the construction of LIT-SiGamal.

Proposition 2. *LIT-SiGamal is OW-CPA secure if the Computational LIT-SiGamal assumption holds.*

Definition 6 (Decisional LIT-SiGamal assumption)**.** We call the following assumption the *Decisional LIT-SiGamal assumption*:

Any PPT algorithm $\mathcal{A}$ satisfies

$$ \left| \Pr\left[i = i^* \;\middle|\; \begin{array}{l} i \xleftarrow{\$} \{0,1\},\ \mu_0 = 1,\ \mu_1 \xleftarrow{\$} (\mathbb{Z}/\ell_C^c\mathbb{Z})^\times, \\ T_{\text{LIT-SiGamal}} \xleftarrow{\$} \mathcal{S}_{\text{LIT-SiGamal}}, \\ i^* \leftarrow \mathcal{A}\left(T_{\text{LIT-SiGamal}}(\mu_i)\right) \end{array} \right] - \frac{1}{2} \right| < \text{negl}(\lambda). $$

We also state the following proposition, whose proof follows in a straightforward manner.

Proposition 3. *LIT-SiGamal is IND-CPA secure if the Decisional LIT-SiGamal assumption holds.*

By Proposition 3, it suffices to analyze the Decisional LIT-SiGamal assumption to evaluate the security of LIT-SiGamal. To this end, we first introduce the LIT-DDH assumption, which serves as the analogue of the classical DDH assumption in the context of a LIT diagram.

Definition 7 (LIT-DDH assumption)**.** Let E_s and E_1 be supersingular elliptic curves with an isogeny $\phi_1 \colon E_s \to E_1$ of degree $\ell_A^{2a} - n^2$. Let $\mathcal{B}_B = (P_B, Q_B)$ be an ordered basis of $E_s[\ell_B^b]$, and let $\mathcal{B}_{A,C}$ be an ordered basis of $E_s[\ell_A^a \ell_C^c]$. Let $\mathbf{A}$ and $\mathbf{B}$ be matrices in $\mathcal{M}_{a,c}$. Define

$$ {}^t\mathcal{B}_{1,A,C} := \mathbf{A} \cdot {}^t\phi_1(\mathcal{B}_{A,C}), \quad \mathcal{B}_{1,B} := \phi_1(\mathcal{B}_B). $$

Let s be an element in $(\mathbb{Z}/\ell_B^b\mathbb{Z})^\times$, let $\phi_2 \colon E_s \to E_s'$ be a separable isogeny with $\ker \phi_2 = \langle P_B + sQ_B \rangle$, and let $\phi_2' \colon E_1 \to E_1'$ be a separable isogeny with $\ker \phi_2' = \langle P_{1,B} + sQ_{1,B} \rangle$, where we set $(P_{1,B}, Q_{1,B}) := \mathcal{B}_{1,B}$. Define

$$ {}^t\mathcal{B}'_{A,C} := \mathbf{B} \cdot {}^t\phi_2(\mathcal{B}_{A,C}), \quad {}^t\mathcal{B}'_{1,A,C} := \mathbf{B} \cdot {}^t\phi_2'(\mathcal{B}_{1,A,C}). $$

We denote by $T_{\text{LIT-DDH},0}$ the tuple

$$ ((E_s, \mathcal{B}_{A,C}, \mathcal{B}_B), (E_1, \mathcal{B}_{1,A,C}, \mathcal{B}_{1,B}), (E_s', \mathcal{B}'_{A,C}), (E_1', \mathcal{B}'_{1,A,C})), $$

and denote by $\mathcal{S}_{\text{LIT-DDH}}$ the set of the above tuples. Let E_r be a supersingular elliptic curve, and let $\mathcal{B}_r$ be an ordered basis of $E_r[\ell_A^a \ell_C^c]$ satisfying $e(P_r, Q_r) = e(P_{1,A,C}, Q_{1,A,C})^{\ell_B^b \det \mathbf{B}}$, where $(P_r, Q_r) = \mathcal{B}_r$ and $(P_{1,A,C}, Q_{1,A,C}) = \mathcal{B}_{1,A,C}$. We denote by $T_{\text{LIT-DDH},1}$ the tuple

$$ ((E_s, \mathcal{B}_{A,C}, \mathcal{B}_B), (E_1, \mathcal{B}_{1,A,C}, \mathcal{B}_{1,B}), (E_s', \mathcal{B}'_{A,C}), (E_r, \mathcal{B}_r)), $$

and denote by $\mathcal{S}_{\text{LIT-DDH}'}$ the set of the above tuples.

We call the following assumption the *LIT-DDH assumption*:

Any PPT algorithm $\mathcal{A}$ satisfies

$$ \left| \Pr \left[i = i^* \;\middle|\; \begin{array}{l} T_{\text{LIT-DDH},0} \xleftarrow{\$} \mathcal{S}_{\text{LIT-DDH}}, \\ T_{\text{LIT-DDH},1} \xleftarrow{\$} \mathcal{S}_{\text{LIT-DDH}'}, \\ i \xleftarrow{\$} \{0,1\},\ i^* \leftarrow \mathcal{A}(T_{\text{LIT-DDH},i}) \end{array} \right] - \frac{1}{2} \right| < \mathrm{negl}(\lambda). $$

We now state the following theorem, whose detailed proof is provided in Appendix A.

Theorem 2. *If the LIT-DDH assumption holds, then the Decisional LIT-SiGamal assumption also holds.*

Combining Proposition 3 and Theorem 2, we obtain the following corollary:

Corollary 1. *If the LIT-DDH assumption holds, LIT-SiGamal is IND-CPA secure.*

Remark 6. LIT-SiGamal is not IND-CCA secure for the same reason as the original SiGamal (see [27, Section 3.3]). Specifically, given a ciphertext

$$\mathbf{ct} = (E'_s, \mathcal{B}_2, R', E'_1, \mathcal{B}_3, R'_1),$$

one can construct a modified ciphertext

$$(E'_s, \mathcal{B}_2, R', E'_1, \mathcal{B}_3, \mu' R'_1)$$

that decrypts to a message scaled by a factor of μ' relative to the original plaintext. In addition, LIT-SiGamal is vulnerable to an adaptive attack, which follows the same strategy as the one proposed in [28].

Therefore, to enable practical use of LIT-SiGamal, an appropriate transformation is necessary. In particular, applying the Fujisaki-Okamoto transform [22], LIT-SiGamal can be transformed into an IND-CCA secure KEM (see Algorithm 1). In this algorithm, the public key encryption scheme $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ refers to the original LIT-SiGamal PKE scheme.

Algorithm 1 Transformed LIT-SiGamal ($\mathsf{KEM} = (\mathsf{KeyGen}, \mathsf{Encaps}, \mathsf{Decaps})$)

function $\mathsf{KeyGen}(\lambda)$
  $r \xleftarrow{\$} \{0,1\}^\lambda$
  $(\mathbf{pk}, \mathbf{sk}) \leftarrow \mathsf{Gen}(\lambda)$
  return $(r, \mathbf{pk}, \mathbf{sk})$
end function

function $\mathsf{Encaps}(\mathbf{pk})$
  $\mu \xleftarrow{\$} (\mathbb{Z}/\ell_C^c\mathbb{Z})^\times$
  $\mathbf{B}, s, \beta \leftarrow \mathsf{Hash}_1(\mu \| \mathbf{pk})$
  $c \leftarrow \mathsf{Enc}(\mathbf{pk}, \mu; \mathbf{B}, s, \beta)$
  $K_S \leftarrow \mathsf{Hash}_2(\mu \| c)$
  return (c, K_S)
end function

function $\mathsf{Decaps}((r, \mathbf{pk}, \mathbf{sk}), c)$
  $\mu' \leftarrow \mathsf{Dec}(\mathbf{sk}, c)$
  $\mathbf{B}', s', \beta' \leftarrow \mathsf{Hash}_1(\mu' \| \mathbf{pk})$
  $c' \leftarrow \mathsf{Enc}(\mathbf{pk}, \mu'; \mathbf{B}', s', \beta')$
  if $c' = c$ **then**
    $K_S \leftarrow \mathsf{Hash}_2(\mu' \| c)$
  else
    $K_S \leftarrow \mathsf{Hash}_2(r \| c)$
  end if
  return K_S
end function

5.2 Secure prime of LIT-SiGamal

In this subsection, we discuss how to choose an appropriate size for the prime p. Recall that p is defined as $p = \ell_A^a \cdot \ell_B^b \cdot \ell_C^c \cdot f - 1$, where f is a small integer. As such, to estimate the bit length of p, it suffices to analyze the sizes of the terms ℓ_A^a, ℓ_B^b, and ℓ_C^c. We denote the security parameter by λ.

We begin by determining the size of ℓ_C^c, which is directly related to the size of the plaintext space. From the definition of the plaintext space $\mathcal{P}$, we have $\#\mathcal{P} \approx \ell_C^c$. Since the plaintext space is desired to have size approximately 2^λ, it follows that $\ell_C^c \approx 2^\lambda$.

From Corollary 1, the security of LIT-SiGamal relies on the LIT-DDH assumption. We begin by considering a direct approach to this decisional assumption, focusing on an algorithm that distinguishes between the tuples $T_{\text{LIT-DDH},0}$ and $T_{\text{LIT-DDH},1}$. Several studies have addressed decisional problems based on instances of the Isogeny Problem (e.g., [12,10,9]). A key tool in these approaches is the use of self-pairings for elliptic curves, which are bilinear pairings $e(\cdot,\cdot)$ such that both input points are the same. A non-trivial self-pairing – one for which $e(P,P) \neq 1$ for some P – can reveal a part of the degree of the hidden isogenies. Specifically, since a self-pairing satisfies $e(\phi(P),\phi(P)) = e(P,P)^{\deg\phi}$, it can expose part of the degree of the isogeny ϕ.

Using this property, previous studies have constructed algorithms to distinguish between two tuples in decisional problems. However, these approaches do not appear to be applicable in the LIT-SiGamal setting. The primary reason is the lack of an efficient method to construct a non-trivial self-pairing for random supersingular elliptic curves over $\mathbb{F}_{p^2}$. To our knowledge, constructing such a self-pairing requires finding a non-scalar endomorphism of a supersingular elliptic curve (see [7, IX.7.2 and IX.7.3]). However, this procedure is as difficult as solving the Isogeny Problem, as shown in the result by Page and Wesolowski [30]. Therefore, these existing methods would require solving problems associated with the Computational LIT-SiGamal assumption. As a result, it is sufficient to focus on computational problems in this context.

We now turn our attention to attempts at solving the LIT and CIST problems (see Definitions 4 and 3). To assess the size of ℓ_A^a and ℓ_B^b, we analyze these problems in the context of LIT-SiGamal.

The size of ℓ_B^b: We now discuss the size of ℓ_B^b. Bob computes ℓ_B^b-isogenies and must ensure that these isogenies are not revealed.
The difficulty of revealing Bob's isogenies is closely related to the hardness of the CIST problem. According to the definition of the CIST problem, information about torsion points is concealed by appropriate matrices. Therefore, we expect that an adversary will not be able to extract valid torsion point information to reveal the isogenies. As a result, the size of ℓ_B^b is derived from the complexity of solving the Isogeny problem of degree ℓ_B^b, which leads us to set $\ell_B^b \approx 2^{2\lambda}$.

The size of ℓ_A^a: We finally estimate an appropriate size for ℓ_A^a. Alice computes an isogeny of degree $\ell_A^{2a} - n^2$ and must ensure that this isogeny remains hidden.
The hardness of revealing Alice's isogeny is related to the hardness of the *LIT problem*. In the scheme, Alice reveals the exact image of a basis of the ℓ_B^b-torsion subgroup under her isogeny. Based on the discussion in [26, Section 4.2], it is suggested to choose ℓ_A^a such that $(\ell_A^{2a} - n^2) > \ell_B^{2b} \cdot 2^{2\lambda}$. Since n is a small integer, it suffices to take ℓ_A^a such that $\ell_A^{2a} \approx \ell_B^{2b} \cdot 2^{2\lambda}$. From the previous discussion, we have $\ell_B^b \approx 2^{2\lambda}$, and thus we set $\ell_A^a \approx \ell_B^b \cdot 2^{\lambda} \approx 2^{3\lambda}$.

Consequently, we suggest to use a prime p about $2^{\lambda} \cdot 2^{2\lambda} \cdot 2^{3\lambda} = 2^{6\lambda}$ for LIT-SiGamal. We provide explicit primes for security parameter $\lambda = 128$, 192, and 256 in Section 6.2.

5.3 LIT-SiGamal v.s. POKÉ

POKÉ is an isogeny-based public key encryption (PKE) scheme proposed by Basso [5]. As noted in Section 1.2, POKÉ shares several structural similarities with LIT-SiGamal. In fact, the constructions of POKÉ and LIT-SiGamal are nearly identical, with the key distinction being the degree of the isogeny used in the public key: POKÉ employs an isogeny of degree approximately $2^{2\lambda}$, whereas LIT-SiGamal uses one of degree approximately $2^{6\lambda}$.

In this subsection, we explore the relationship between the security of these two schemes. We do not believe POKÉ is currently vulnerable; however, since both schemes are relatively new, we consider some kind of security reduction between these schemes worth careful consideration. We present a reduction between the mathematical problems underlying their respective security. While LIT-SiGamal (specifically, the variant in which the degree of the public key isogeny ϕ_1 is hidden; see Remark 2) appears to offer stronger security than POKÉ, this is largely because LIT-SiGamal can be interpreted as a strengthened version of POKÉ that uses a higher-degree isogeny for ϕ_1. Notably, an adversary may be able to break POKÉ if it knows $\deg(\phi_1)$, while the same information may not suffice to break LIT-SiGamal.

Our ultimate goal is to provide a formal reduction supporting the above intuition—that is, to identify a reduction between problems that capture the respective security of POKÉ and LIT-SiGamal. However, constructing such a reduction directly between the actual schemes is challenging due to minor technical differences. For example, in LIT-SiGamal, the public key isogeny has degree of the form $2^{2a} - n^2$, whereas in POKÉ, it takes the form $(2^a - q)q$ for some integer $q \in \mathbb{Z}$.

Therefore, instead of working with the concrete schemes, we focus on "idealized" versions of the associated problems and analyze a reduction between them, as defined below.

Definition 1 (Idealized LIT-SiGamal problem). *Let p be a prime, and let A and B be smooth integers such that $A \approx p^{1/2}$, $B \approx p^{1/3}$, $\gcd(A, B) = 1$, and $AB|(p+1)$. Let E and E' be supersingular elliptic curves over $\mathbb{F}_{p^2}$, and let C_1 and C_2 (resp. C_1' and C_2') be cyclic subgroups of E (resp. E') of order A satisfying $C_1 + C_2 = E[A]$ (resp. $C_1' + C_2' = E[A]$). Let O_B be an oracle that outputs $\langle \phi(P) \rangle \subset E'[B]$ from $P \in E[B]$. We put $\mathcal{E} = (E, C_1, C_2)$ and $\mathcal{E}' = (E', C_1', C_2')$.*

We define the idealized LIT-SiGamal problem for (p, A, B) as the following problem:

> *Find a representation of a cyclic isogeny $\phi\colon \mathcal{E} \to \mathcal{E}'$ Let $\phi\colon E \to E'$ with $\deg \phi \approx p$ and $\gcd(\deg \phi, AB) = 1$, from $(\mathcal{E}, \mathcal{E}', O_B)$.*

Definition 2 (Idealized POKÉ problem). *Let p be a prime, and let A and B be smooth integers such that $A \approx p^{1/3}$, $B \approx p^{2/3}$, $\gcd(A, B) = 1$, and $AB|(p+1)$. Let E and E' be supersingular elliptic curves over $\mathbb{F}_{p^2}$, and let C_1 and C_2 (resp. C_1' and C_2') be cyclic subgroups of E (resp. E') of order A satisfying $C_1 + C_2 = E[A]$ (resp. $C_1' + C_2' = E[A]$). Let O_B be an oracle that outputs $\langle \phi(P) \rangle \subset E'[B]$ from $P \in E[B]$. We put $\mathcal{E} = (E, C_1, C_2)$ and $\mathcal{E}' = (E', C_1', C_2')$.*

Table 3. The parameters for the idealized LIT-SiGamal and POKÉ problems

	$\deg\phi$	A	B
LIT-SiGamal	$\approx p$	$\approx p^{1/2}$	$\approx p^{1/3}$
POKÉ	$\approx p^{2/3}$	$\approx p^{1/3}$	$\approx p^{2/3}$

We define the idealized POKÉ problem for (p, A, B) as the following problem:

Find a representation of a cyclic isogeny $\phi\colon \mathcal{E} \to \mathcal{E}'$ Let $\phi\colon E \to E'$ with $\deg\phi \approx p^{2/3}$ and $\gcd(\deg\phi, AB) = 1$, from $(\mathcal{E}, \mathcal{E}', O_B)$.

We now present a reduction between the idealized LIT-SiGamal and POKÉ problems. The following proposition shows that the idealized POKÉ problem can be reduced to the idealized LIT-SiGamal problem under a specific condition. Although this proposition does not establish a concrete reduction between POKÉ and LIT-SiGamal, it provides supporting evidence for the claim that LIT-SiGamal offers stronger security guarantees than POKÉ.

Proposition 4. *Let p be a prime, and let M be a small integer. Let A, B, and B' be M-smooth integers such that $A \approx B \approx B' \approx p^{1/3}$, $ABB' | (p+1)$, and they are pairwise coprime. Let A' be a smooth integer satisfying $A' | AB'$ and $A' \approx p^{1/2}$.*

Assume that there is an algorithm $\mathcal{A}$ that solves the idealized LIT-SiGamal problem for (p, A', B) in time T. Then, there exists an algorithm that solves the idealized POKÉ problem for (p, A, BB') in time $T + \mathrm{poly}(\log p, M)$, where $\mathrm{poly}(\log p, M)$ denotes a time complexity that is polynomial in $\log p$ and M.

Proof. Let C_1 and C_2 be cyclic subgroups of $E[A]$ of order A such that $C_1 + C_2 = E[A]$, let C_1' and C_2' be cyclic subgroups of $E'[A]$ of order A such that $C_1' + C_2' = E'[A]$, and let $O_{BB'}$ be an oracle that outputs $\langle\phi(P)\rangle$ from $P \in E[BB']$. We put $\mathcal{E} = (E, C_1, C_2)$ and $\mathcal{E}' = (E', C_1', C_2')$. Our goal is to construct an algorithm $\mathcal{A}'$ that solves the idealized POKÉ problem in time polynomial in $T + \mathrm{poly}(\log p, M)$. In other words, we construct an algorithm $\mathcal{A}'$ that outputs an isogeny $\phi\colon \mathcal{E} \to \mathcal{E}'$, which is a cyclic isogeny satisfying $\deg\phi \approx p^{2/3}$ and $\deg\phi$ is coprime to ABB', from $(\mathcal{E}, \mathcal{E}', O_{BB'})$.

Assume that we have a cyclic isogeny $\phi'\colon E' \to E''$ with $\deg\phi' \approx p^{1/3}$ and $\gcd(\deg\phi', ABB'\deg\phi) = 1$. Since $\phi' \circ \phi$ is a cyclic isogeny of degree $\approx p$, we can construct the algorithm $\mathcal{A}'$ as follows:

1. Take cyclic subgroups C_1' and C_2' of $E[AB']$ of order AB' such that $C_1' + C_2' = E[AB']$ and $C_i \subset C_i'$ for $i = 1, 2$. Note that, from the Chinese remainder theorem, C_i' is generated by a generator of C_i and a point $P_{B',i}$ of order B'.
2. Define C_i'' by $C_i'' := (AB'/A')C_i'$ for $i = 1, 2$.
3. Compute $(\phi' \circ \phi)(C_i'')$ for $i = 1, 2$ from ϕ', $\phi(C_i)$, $P_{B',i}$, and $O_{BB'}$.
4. Define an oracle O_B by $O_B(P) := \phi'(O_{BB'}(P))$ for $P \in E[B]$.
5. Obtain $\phi' \circ \phi$ by $\mathcal{A}((E, C_1'', C_2''), (E'', (\phi' \circ \phi)(C_1''), (\phi' \circ \phi)(C_2'')), O_B)$.
6. Compute ϕ from $\phi' \circ \phi$ and ϕ'.

Therefore, the proposition holds if we can compute ϕ' in $\mathrm{poly}(\log p, M)$.

We now explain the way to compute ϕ' in polynomial time. Let d be a prime with $\gcd(d, ABC\deg\phi) = 1$ and $d \approx M$. We assume that $\deg\phi' = d^e \approx p^{1/3}$ with $e = \lceil \log_d p^{1/3} \rceil$, and there is a decomposition of ϕ' with $\phi' = \phi'_e \circ \cdots \circ \phi'_1$, where $\phi'_1, \ldots, \phi'_e$ are d-isogenies. We denote the codomain of ϕ'_i as E'_i. By using the modular polynomial level d, we can compute a d-isogeny $\phi'_1 \colon E' \to E'_1$ in time polynomial in $\log p$ and M. To compute image points under ϕ'_1, we use a technique for Robert's attack [34]. We first take an ordered basis $\mathcal{B}_A$ in $E'[A_d]$, where A_d is the smallest integer satisfying $A_d | A$ and $A_d > \sqrt{d}$. We next take all bases of $E'_1[A_d]$ and see in which basis Robert's attack works for revealing ϕ'_1. Robert's attack works if and only if we take n ordered basis $\mathcal{B}'_A$ that corresponds to the image of $\mathcal{B}_A$ under a d-isogeny $\phi''_1 \colon E' \to E'_1$. Note that ϕ''_1 may not be ϕ'_1, but we can replace ϕ'_1 by ϕ''_1. Thus, we can compute $\phi'_1(\mathcal{B}_A)$ in time polynomial in $\log p$ and M. Since $(\mathcal{B}_A, \phi'_1(\mathcal{B}_A))$ is a representation of ϕ'_1, we can compute any image point under ϕ'_1. Using this strategy, we can also compute $\phi'_2, \ldots, \phi'_e$. Therefore, we can compute ϕ' in polynomial time by repeating the above procedure e times. □

6 Performance of LIT-SiGamal

In this section, we discuss the performance of LIT-SiGamal.

6.1 High-level analysis of the computational costs of LIT-SiGamal

In this subsection, we provide a high-level analysis of the computational costs associated with LIT-SiGamal. Specifically, we analyze the costs of the encryption and decryption algorithms of LIT-SiGamal. We then compare these costs with those of other isogeny-based schemes, including POKÉ, QFESTA, and SIDH. Although SIDH has been broken by the Castryck–Decru attack, we include it here as a point of reference for performance comparisons. A summary of our analysis is presented in Table 4.

Let λ denote the security parameter. We define the cost of a multiplication over $\mathbb{F}_{p^2}$ as $\mathbf{M}_{\mathsf{bit}(p)}$, where $\mathsf{bit}(p)$ represents the bit length of p.

Encryption: The most computationally intensive part of the LIT-SiGamal encryption algorithm is the calculation of two ℓ_B^b-isogenies. Therefore, the cost of encryption is given by

$$O(\log_2(\ell_B^b) \cdot \log_2(\log_2(\ell_B^b)) \cdot \mathbf{M}_{\mathsf{bit}(p)}) = O(\lambda \log\lambda \cdot \mathbf{M}_{6\lambda}).$$

Here, we use the approximations $\ell_B^b \approx 2^{2\lambda}$ and $p \approx 2^{6\lambda}$, as discussed in Section 5.2.

Decryption: The most computationally intensive part of the LIT-SiGamal decryption algorithm is the calculation of an (ℓ_A^a, ℓ_A^a)-isogeny. Therefore, the cost of decryption is given by

$$O(\log_2(\ell_A^a) \cdot \log_2(\log_2(\ell_A^a)) \cdot \mathbf{M}_{\mathsf{bit}(p)}) = O(\lambda \log\lambda \cdot \mathbf{M}_{6\lambda}).$$

Table 4. Computational costs of some isogeny-based schemes

	LIT-SiGamal	POKÉ [5]	QFESTA [29]	SIDH [18]
Encryption	$\lambda \log \lambda \cdot \mathbf{M}_{6\lambda}$	$\lambda \log \lambda \cdot \mathbf{M}_{3\lambda}$	$\lambda^2 \cdot \mathbf{M}_{3\lambda}$	$\lambda \log \lambda \cdot \mathbf{M}_{4\lambda}$
Decryption	$\lambda \log \lambda \cdot \mathbf{M}_{6\lambda}$	$\lambda \log \lambda \cdot \mathbf{M}_{3\lambda}$	$\lambda \log \lambda \cdot \mathbf{M}_{3\lambda}$	$\lambda \log \lambda \cdot \mathbf{M}_{4\lambda}$
Encapsulation	$\lambda \log \lambda \cdot \mathbf{M}_{6\lambda}$	$\lambda \log \lambda \cdot \mathbf{M}_{3\lambda}$	$\lambda^2 \cdot \mathbf{M}_{3\lambda}$	$\lambda \log \lambda \cdot \mathbf{M}_{4\lambda}$
Decapsulation	$\lambda \log \lambda \cdot \mathbf{M}_{6\lambda}$	$\lambda \log \lambda \cdot \mathbf{M}_{3\lambda}$	$\lambda^2 \cdot \mathbf{M}_{3\lambda}$	$\lambda \log \lambda \cdot \mathbf{M}_{4\lambda}$

Here, we use the approximation $\ell_A^a \approx 2^{3\lambda}$.

The construction of POKÉ is similar to that of LIT-SiGamal. The most computationally intensive part of its encryption algorithm involves computing two isogenies of degree approximately $2^{2\lambda}$, while the most computationally intensive part of its decryption algorithm involves computing a $(2^\lambda, 2^\lambda)$-isogeny. Since POKÉ operates over $\mathbb{F}_{p^2}$ with $p \approx 2^{3\lambda}$, the computational cost of the POKÉ encryption and decryption algorithms is $O(\lambda \log \lambda \cdot \mathbf{M}_{3\lambda})$.

The QFESTA decryption algorithm is quite similar to that of POKÉ, and therefore, its computational cost is $O(\lambda \log \lambda \cdot \mathbf{M}_{3\lambda})$. The most computationally intensive part of the QFESTA encryption algorithm is the calculation of an isogeny of degree approximately $2^{2\lambda}$, which is performed using a radical isogeny formula. Note that the cost of this computation is $O(2\lambda \cdot \mathsf{bit}(p) \cdot \mathbf{M}_p)$. Consequently, the total computational cost of the QFESTA encryption algorithm is $O(\lambda^2 \cdot \mathbf{M}_{3\lambda})$.

SIDH is primarily a key exchange protocol, not a public key encryption (PKE) scheme. For the purposes of this discussion, we assume that SIDH can be transformed into a PKE scheme using a hash function and the XOR operator. The most computationally intensive part of the SIDH encryption and decryption algorithms involves computing two isogenies of degree approximately $2^{2\lambda}$. Therefore, the computational cost of these operations is $O(\lambda \log \lambda \cdot \mathbf{M}_{4\lambda})$, since SIDH operates over $\mathbb{F}_{p^2}$ with $p \approx 2^{4\lambda}$.

We summarize the above discussion in Table 4. Note that, as shown in Algorithm 1, the cost of encapsulation is the same as that of encryption, and the cost of decapsulation is the sum of the costs of encryption and decryption. Although we have ignored constant factors in the above analysis, we conclude that POKÉ and SIDH are likely to be more efficient than LIT-SiGamal due to their lower multiplication costs. However, QFESTA is expected to be less efficient than LIT-SiGamal. Consequently, LIT-SiGamal is expected to be more efficient than QFESTA, but less efficient than POKÉ and SIDH.

6.2 Parameter selection and the size of the scheme

In this subsection, we propose primes for LIT-SiGamal corresponding to the security parameters of 128, 192, and 256, based on the discussion in Section 5.2. Additionally, we provide the sizes of the public keys and ciphertexts of LIT-SiGamal when using these primes, and compare them with those of other isogeny-based public key encryption (PKE) schemes.

Table 5. The sizes of LIT-SiGamal (byte)

	Public key		Ciphertext	
λ	Original	Compressed	Original	Compressed
128	$2,334$	664	$1,556$	728
192	$3,489$	992	$2,326$	$1,088$
256	$4,653$	$1,322$	$3,102$	$1,451$

Table 6. Comparison of the sizes of isogeny-based PKEs (byte)

	λ	LIT-SiGamal	FESTA [6]	QFESTA [29]	terSIDH^{hyb} [4]	POKÉ [5]
Public key	128	664	561	247	701	272
	192	991	864	367	$1,089$	408
	256	$1,322$	$1,246$	487	$1,479$	544
Ciphertext	128	728	$1,122$	494	459	384
	192	$1,088$	$1,728$	734	706	576
	256	$1,451$	$2,492$	974	956	768

We denote by p_λ the prime corresponding to the security parameter λ. We define the primes as follows:

$$p_{128} = 2^{128\cdot 3+2} \cdot 3^{162} \cdot 5^{56} \cdot 30 - 1,$$
$$p_{192} = 2^{192\cdot 3+2} \cdot 3^{243} \cdot 5^{83} \cdot 118 - 1,$$
$$p_{256} = 2^{256\cdot 3+2} \cdot 3^{324} \cdot 5^{111} \cdot 436 - 1.$$

The "+2" in the exponents is used for computing isogenies between abelian varieties of dimension 2 via theta coordinates (see [17]). The bit length of p_{128} is 778, that of p_{192} is 1163, and that of p_{256} is 1551.

Table 5 summarizes the sizes of the public keys and ciphertexts of LIT-SiGamal using these primes. Here, the compressed LIT-SiGamal refers to the variation of LIT-SiGamal described in Remark 4. Additionally, Table 6 compares the sizes of the public keys and ciphertexts of compressed LIT-SiGamal with those of several other isogeny-based PKE schemes. Here, we assume that terSIDH^{hyb} [4] is transformed into a PKE scheme by using the XOR operator and a hash function. As shown in Table 6, the public key of the compressed LIT-SiGamal is larger than that of FESTA, and its ciphertext is smaller than that of FESTA.

6.3 Rust implementation

We have developed a Rust implementation that is both reusable and tailored to support components required for isogeny-based cryptographic protocols, including those underlying LIT-SiGamal. Our implementation builds on several existing projects:

– The `two-isogenies` Rust library for $(2^e, 2^e)$-isogenies, as introduced in [17].

Table 7. Computational time for Rust implementation of LIT-SiGamal (sec.)

λ	Public key gen.	Encryption	Decryption
128	1.33	0.03	0.09
192	4.62	0.10	0.32
256	11.28	0.24	0.78

- The `two-isogenies` library itself depends on Pornin's `crrl` library[5] for finite field arithmetic and elliptic curve operations.
- The quaternion algebra library from [31].

While several challenges remain—such as achieving constant-time execution, particularly for the Cornacchia operation—our library offers a range of useful tools for isogeny-based cryptography.

There is also room for significant performance improvements, especially by incorporating faster discrete logarithm routines. Our Rust implementation is available at: `https://github.com/LIT-SiGamal/LIT-SiGamal`.

Table 7 presents the experimental results of our Rust implementation. The benchmarks were conducted on an Apple M1 MacBook Pro (3.2 GHz), measuring the computational time.

7 Conclusion

We propose a novel isogeny-based public-key encryption (PKE) scheme named LIT-SiGamal. This scheme builds upon the SiGamal PKE scheme but replaces the CSIDH diagram with a LIT diagram. Additionally, we present a compressed variant of LIT-SiGamal that leverages the compression techniques used in SIDH.

We prove that LIT-SiGamal achieves IND-CPA security under the LIT-DDH assumption, which is analogous to the classical DDH assumption, but in the context of a LIT diagram. Furthermore, we identify the possibility of an adaptive attack against the scheme. We also compare the security of LIT-SiGamal with that of POKÉ. Although we do not provide a formal proof, LIT-SiGamal appears to offer stronger security guarantees than POKÉ.

The computational complexities of both the encryption and decryption procedures in LIT-SiGamal are given by $O(\lambda \log \lambda \cdot \mathbf{M}_{6\lambda})$ for a security parameter λ. This analysis suggests that LIT-SiGamal is more efficient than QFESTA.

Finally, to assess the practicality of our scheme, we implemented LIT-SiGamal in Rust. The current implementation achieves reasonable performance, although there remains substantial room for further optimization.

Acknowledgements. We thank the anonymous reviewers for their helpful feedback and constructive comments, which significantly improved the quality of this work. We also made use of a generative AI tool to assist with refining the English writing.

This work was supported by EPSRC through grant EP/V011324/1.

[5] `https://github.com/pornin/crrl`

References

1. Noga Alon, Itai Benjamini, Eyal Lubetzky, and Sasha Sodin. Non-backtracking random walks mix faster. *Communications in Contemporary Mathematics*, 9(04):585–603, 2007.
2. Sarah Arpin. Adding level structure to supersingular elliptic curve isogeny graphs. *Journal de Théorie des Nombres de Bordeaux*, 36(2):405–443, 2024.
3. Andrea Basso, Giulio Codogni, Deirdre Connolly, Luca De Feo, Tako Boris Fouotsa, Guido Maria Lido, Travis Morrison, Lorenz Panny, Sikhar Patranabis, and Benjamin Wesolowski. Supersingular curves you can trust. In *Advances in Cryptology – EUROCRYPT 2023*, pages 405–437. Springer, 2023.
4. Andrea Basso and Tako Boris Fouotsa. New SIDH countermeasures for a more efficient key exchange. In *Advances in Cryptology – ASIACRYPT 2023*, pages 208–233. Springer, 2023.
5. Andrea Basso and Luciano Maino. POKÉ: A compact and efficient PKE from higher-dimensional isogenies, 2024. `https://eprint.iacr.org/2024/624` appear at EUROCRYPT 2025.
6. Andrea Basso, Luciano Maino, and Giacomo Pope. FESTA: Fast encryption from supersingular torsion attacks. In *Advances in Cryptology – ASIACRYPT 2023*, pages 98–126. Springer, 2023.
7. Ian F Blake, Gadiel Seroussi, and Nigel P Smart. *Advances in elliptic curve cryptography*, volume 2. Cambridge University Press, 2005.
8. Wouter Castryck and Thomas Decru. An efficient key recovery attack on SIDH. In *Advances in Cryptology – EUROCRYPT 2023*, pages 423–447. Springer, 2023.
9. Wouter Castryck, Marc Houben, Simon-Philipp Merz, Marzio Mula, Sam van Buuren, and Frederik Vercauteren. Weak instances of class group action based cryptography via self-pairings. In *Annual International Cryptology Conference – CRYPTO 2023*, pages 762–792. Springer, 2023.
10. Wouter Castryck, Marc Houben, Frederik Vercauteren, and Benjamin Wesolowski. On the decisional Diffie–Hellman problem for class group actions on oriented elliptic curves. *Research in Number Theory*, 8(4):99, 2022.
11. Wouter Castryck, Tanja Lange, Chloe Martindale, Lorenz Panny, and Joost Renes. CSIDH: an efficient post-quantum commutative group action. In *Advances in Cryptology – ASIACRYPT 2018*, pages 395–427. Springer, 2018.
12. Wouter Castryck, Jana Sotáková, and Frederik Vercauteren. Breaking the decisional Diffie-Hellman problem for class group actions using genus theory. In *Annual International Cryptology Conference – CRYPTO 2020*, pages 92–120. Springer, 2020.
13. Wouter Castryck and Frederik Vercauteren. A polynomial time attack on instances of M-SIDH and FESTA. In *Advances in Cryptology – ASIACRYPT 2023*, pages 127–156. Springer, 2023.
14. Jorge Chávez-Saab, Jesús-Javier Chi-Domínguez, Samuel Jaques, and Francisco Rodríguez-Henríquez. The SQALE of CSIDH: sublinear vélu quantum-resistant isogeny action with low exponents. *Journal of Cryptographic Engineering*, 12(3):349–368, 2022.
15. Craig Costello, David Jao, Patrick Longa, Michael Naehrig, Joost Renes, and David Urbanik. Efficient compression of SIDH public keys. In *Advances in Cryptology – EUROCRYPT 2017*, pages 679–706. Springer, 2017.
16. Pierrick Dartois, Antonin Leroux, Damien Robert, and Benjamin Wesolowski. SQISignHD: new dimensions in cryptography. In *Advances in Cryptology – EUROCRYPT 2024*, pages 3–32. Springer, 2024.

17. Pierrick Dartois, Luciano Maino, Giacomo Pope, and Damien Robert. An algorithmic approach to $(2, 2)$-isogenies in the theta model and applications to isogeny-based cryptography. In *International Conference on the Theory and Application of Cryptology and Information Security – ASIACRYPT 2024*, pages 304–338. Springer, 2024.
18. Luca De Feo, David Jao, and Jérôme Plût. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. *Journal of Mathematical Cryptology*, 8(3):209–247, 2014.
19. Luca De Feo, David Kohel, Antonin Leroux, Christophe Petit, and Benjamin Wesolowski. SQISign: compact post-quantum signatures from quaternions and isogenies. In *Advances in Cryptology – ASIACRYPT 2020*, pages 64–93. Springer, 2020.
20. The Sage Developers. Sagemath, version 10.4, 2023. `http://www.sagemath.org`.
21. Tako Boris Fouotsa, Tomoki Moriya, and Christophe Petit. M-SIDH and MD-SIDH: Countering SIDH attacks by masking information. In *Advances in Cryptology – EUROCRYPT 2023*, pages 282–309. Springer, 2023.
22. Eiichiro Fujisaki and Tatsuaki Okamoto. Secure integration of asymmetric and symmetric encryption schemes. In *Advances in Cryptology – CRYPTO'99*, pages 537–554. Springer, 1999.
23. David Jao and Luca De Feo. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. In *Post-Quantum Cryptography – PQCrypto 2011*, pages 19–34. Springer, 2011.
24. Ernst Kani. The number of curves of genus two with elliptic differentials. 1997.
25. Luciano Maino, Chloe Martindale, Lorenz Panny, Giacomo Pope, and Benjamin Wesolowski. A direct key recovery attack on SIDH. In *Advances in Cryptology – EUROCRYPT 2023*, pages 448–471. Springer, 2023.
26. Tomoki Moriya. IS-CUBE: An isogeny-based compact KEM using a boxed SIDH diagram, 2023. `https://eprint.iacr.org/2023/1506`.
27. Tomoki Moriya, Hiroshi Onuki, and Tsuyoshi Takagi. SiGamal: A supersingular isogeny-based pke and its application to a prf. In *Advances in Cryptology – ASIACRYPT 2020*, pages 551–580. Springer, 2020.
28. Tomoki Moriya, Hiroshi Onuki, Maozhi Xu, and Guoqing Zhou. Adaptive attacks against FESTA without input validation or constant-time implementation. In *International Conference on Post-Quantum Cryptography — PQCRYPTO 2024*, pages 3–19. Springer, 2024.
29. Kohei Nakagawa and Hiroshi Onuki. QFESTA: Efficient algorithms and parameters for FESTA using quaternion algebras. In *Annual International Cryptology Conference – CRYPTO 2024*, pages 75–106. Springer, 2024.
30. Aurel Page and Benjamin Wesolowski. The supersingular endomorphism ring and one endomorphism problems are equivalent. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques – EUROCRYPT 2024*, pages 388–417. Springer, 2024.
31. Lorenz Panny, Christophe Petit, and Miha Stopar. Klapoti: an asymptotically efficient isogeny group action from 2-dimensional isogenies. *Cryptology ePrint Archive*, 2024.
32. Stephen Pohlig and Martin Hellman. An improved algorithm for computing logarithms over $GF(p)$ and its cryptographic significance. *IEEE Transactions on information Theory*, 24(1):106–110, 1978.
33. Damien Robert. Efficient algorithms for abelian varieties and their moduli spaces, 2021. Université de Bordeaux (UB). `https://hal.science/tel-03498268`.

34. Damien Robert. Breaking SIDH in polynomial time. In *Advances in Cryptology – EUROCRYPT 2023*, pages 472–503. Springer, 2023.
35. Joseph Silverman. *The arithmetic of elliptic curves*, volume 106. Springer Science & Business Media, 2009.
36. Benjamin Smith. *Explicit endomorphisms and correspondences*. PhD thesis, University of Sydney, 2005.
37. Jacques Vélu. Isogénies entre courbes elliptiques. *CR Acad. Sci. Paris Sér. A*, 273(5):238–241, 1971.

Appendix A Proof of Theorem 2

Here, we give a precise proof of Theorem 2.

Proof. Suppose that the Decisional LIT-SiGamal assumption does not hold. That is, there is a PPT algorithm $\mathcal{A}'$ satisfying the value

$$\varepsilon := \left| \Pr\left[i = i^* \;\middle|\; \begin{array}{l} i \xleftarrow{\$} \{0,1\},\ \mu_0 = 1,\ \mu_1 \xleftarrow{\$} (\mathbb{Z}/\ell_C^c\mathbb{Z})^\times, \\ T_{\text{LIT-SiGamal}} \xleftarrow{\$} \mathcal{S}_{\text{LIT-SiGamal}}, \\ i^* \leftarrow \mathcal{A}'\left(T_{\text{LIT-SiGamal}}(\mu_i)\right) \end{array} \right] - \frac{1}{2} \right|$$

is not negligible.

Let the following tuple T be a tuple in $\mathcal{S}_{\text{LIT-DDH}} \cup \mathcal{S}_{\text{LIT-DDH}'}$:

$$((E_s, \mathcal{B}_{A,C}, \mathcal{B}_B), (E_1, \mathcal{B}_{1,A,C}, \mathcal{B}_{1,B}), (E'_s, \mathcal{B}'_{A,C}), (E_2, \mathcal{B}_2)).$$

From the Chinese remainder theorem, we have an isomorphism between $E[\ell_A^a \ell_C^c]$ and $E[\ell_A^a] \times E[\ell_C^c]$ for any supersingular elliptic curve E. By using these isomorphisms, a basis of $E[\ell_A^a \ell_C^c]$ is mapped to bases of $E[\ell_A^a]$ and $E[\ell_C^c]$. Additionally, by removing one point of order ℓ_C^c, a basis of $E[\ell_A^a \ell_C^c]$ is mapped to a basis of $E[\ell_A^a]$ and a point of order ℓ_C^c. Therefore, we have a transformation from the tuple T to a tuple

$$((E_s, \mathcal{B}_A, \mathcal{B}_B, R), (E_1, \mathcal{B}_{1,A}, \mathcal{B}_{1,B}, R_1), (E'_s, \mathcal{B}'_A, R'), (E_2, \mathcal{B}_{2,A}, R_2)),$$

where $\mathcal{B}_A$ (resp. $\mathcal{B}_{1,A}$, $\mathcal{B}'_A$, $\mathcal{B}_{2,A}$) is a basis of $E_s[\ell_A^a]$ (resp. $E_1[\ell_A^a]$, $E'_s[\ell_A^a]$, $E_2[\ell_A^a]$), and R (resp. R_1, R', R_2) is a point in E_s (resp. E_1, E'_s, E_2) of order ℓ_C^c derived from $P_{A,C}$ with $\mathcal{B}_{A,B} = (P_{A,B}, Q_{A,B})$ (resp. $P_{1,A,C},$, $P'_{A,C}$, P_2). We denote this map by $F_{\text{DDH}\to\text{SiGamal}}$.

We define a PPT algorithm $\mathcal{A}$ with an input T as the following procedure:

1. Take i from $\{0,1\}$ uniformly at random.
2. Set $\mu_0 = 1$ and take μ_1 from $(\mathbb{Z}/\ell_C^c\mathbb{Z})^\times$ uniformly at random.
3. Compute $T' = F_{\text{DDH}\to\text{SiGamal}}(T)$.
4. Compute $i^* \leftarrow \mathcal{A}'(T'(\mu_i))$.
5. if $i = i^*$, output 0, and if $i \neq i^*$, output 1.

Table 8. Computational time of LIT-SiGamal from Sage (sec.)

	Public key gen.		Encryption		Decryption	
λ	Original	Compressed	Original	Compressed	Original	Compressed
128	6.88	8.42	0.12	1.41	0.60	1.08
192	15.97	19.36	0.25	3.35	1.47	2.56
256	29.67	37.99	0.46	6.31	3.06	5.04

Note that if $T \in \mathcal{S}_{\text{LIT-DDH}}$, then $T' \in \mathcal{S}_{\text{LIT-SiGamal}}$. Moreover, if $T \in \mathcal{S}_{\text{LIT-DDH}'}$, then

$$\Pr[1 \leftarrow \mathcal{A}(T)] = \frac{1}{2}$$

because we cannot distinguish $T'(\mu_0)$ and $T'(\mu_1)$. Therefore, it holds that

$$\begin{aligned}
&\Pr\left[\, i = i^* \;\middle|\; \begin{array}{l} T_{\text{LIT-DDH},0} \xleftarrow{\$} \mathcal{S}_{\text{LIT-DDH}}, \\ T_{\text{LIT-DDH},1} \xleftarrow{\$} \mathcal{S}_{\text{LIT-DDH}'}, \\ i \xleftarrow{\$} \{0,1\},\ i^* \leftarrow \mathcal{A}(T_{\text{LIT-DDH},i}) \end{array} \right] \\
&= \frac{1}{2}\Pr[0 \leftarrow \mathcal{A}(T_{\text{LIT-DDH},0})] + \frac{1}{2}\Pr[1 \leftarrow \mathcal{A}(T_{\text{LIT-DDH},1})] \\
&= \frac{1}{2}\left(\frac{1}{2}\Pr[0 \leftarrow \mathcal{A}'(T_{\text{LIT-SiGamal}})] + \frac{1}{2}\Pr[1 \leftarrow \mathcal{A}'(T_{\text{LIT-SiGamal}}(\mu_1))]\right) + \frac{1}{4} \\
&= \frac{1}{2}\left(\frac{1}{2} \pm \varepsilon\right) + \frac{1}{4} = \frac{1}{2} + \frac{\varepsilon}{2}.
\end{aligned}$$

Hence, the algorithm $\mathcal{A}$ can distinguish $T_{\text{LIT-DDH},0}$ and $T_{\text{LIT-DDH},1}$, and the LIT-DDH assumption does not hold. This completes the proof of Theorem 2. □

Appendix B PoC implementation via SageMath

In this section, we present the results related to our proof-of-concept implementation of LIT-SiGamal. We implemented LIT-SiGamal using SageMath [20], and the source code is available at `https://tomoriya.work/code.html`.

Table 8 shows the computational times of the original and compressed versions of LIT-SiGamal. For the underlying prime fields, we used the primes p_{128}, p_{192}, and p_{256} defined in Section 6.2. Each data point represents the average of 100 runs of the corresponding algorithm. The implementation was executed using SageMath version 10.4 on a MacBook Air equipped with an Apple M1 CPU running at 3.2 GHz.

We also measured the average run times of proof-of-concept (PoC) implementations of FESTA [6], QFESTA [29], and terSIDH^{hyb} [4] under the same experimental environment. These results are summarized in Table 9, which compares the computational times across the different schemes. As shown in the table, LIT-SiGamal appears to be more efficient than QFESTA. Furthermore, for security parameters $\lambda = 192$ and 256, the combined encryption and decryption time of LIT-SiGamal is the second most efficient among the compared schemes.

Table 9. Comparison of the computational times of isogeny-based PKEs from Sage (sec.)

	λ	LIT-SiGamal	FESTA [6]	QFESTA [29]	terSIDH^{hyb} [4]
Pub. key	128	8.42	4.16	1.02	2.97
	192	19.36	89.19	2.31	11.51
	256	37.99	272.98	4.51	31.14
Encrypt.	128	1.41	2.85	-	0.38
	192	3.35	17.79	-	1.02
	256	6.31	52.68	-	2.26
Decrypt.	128	1.08	8.69*	-	2.16
	192	2.56	22.04*	-	8.18
	256	5.04	53.69*	-	21.93
Encaps.	128	1.41†	-	1.42	-
	192	3.35†	-	3.36	-
	256	6.31†	-	6.82	-
Decaps.	128	1.20†	-	3.98	-
	192	2.81†	-	10.62	-
	256	5.50†	-	23.11	-

Remark 7. The PoC implementation of QFESTA provides computational times for a key encapsulation mechanism (KEM) derived from QFESTA. Therefore, to enable a fair comparison with LIT-SiGamal, we estimate the computational costs of a KEM based on LIT-SiGamal in Table 9. Specifically, we approximate the encapsulation time of LIT-SiGamal by its encryption time, and the decapsulation time by the sum of the decryption time and the (non-compressed) encryption time.

Remark 8. The decryption process of FESTA can be optimized using algorithms based on theta coordinates, as discussed in [17]. However, the current PoC implementation of FESTA does not incorporate these optimizations and is therefore less efficient than the theoretical state-of-the-art. Consequently, the decryption times for FESTA shown in Table 9 should be interpreted as rough indicators rather than definitive performance measures.

We also note that FESTA is generally considered less efficient than QFESTA, primarily because it operates over a larger prime field.

Remark 9. terSIDH^{hyb} is originally a key exchange protocol, not a public key encryption (PKE) scheme. For comparison, we consider a PKE variant derived from terSIDH^{hyb} using a hash function and the XOR operator. As terSIDH^{hyb} is an asymmetric protocol (i.e., the computational cost differs significantly between the two parties), there are multiple possible constructions for a PKE scheme. In Table 9, we report the computational times for the variant that yields the lowest total cost for encryption and decryption.

Remark 10. As noted in Section 6.1, POKÉ [5] is expected to be more efficient than LIT-SiGamal. This is primarily because the bit length of the prime p used in POKÉ is approximately half that of the prime used in LIT-SiGamal.

Although we aimed to validate this estimation through experimental comparison, we were unable to benchmark POKÉ and LIT-SiGamal under the same environment due to environmental limitations.

Appendix C Computation of $(2,2)$-isogenies via theta coordinates

In the public key generation and decryption of LIT-SiGamal, we need to compute isogenies of high-dimensional varieties. In particular, we need to calculate $(2,2)$-isogenies. There are two well-known methods for the computation of $(2,2)$-isogenies: the method of Richelot isogenies and that of theta coordinates with level 2 structure. We chose the method of theta coordinates for computing these isogenies because this method has been shown to be more efficient than the Richelot isogeny method in practice.

Dartois, Maino, Pope, and Robert provided a concrete algorithm to compute $(2,2)$-isogenies using theta coordinates in [17]. Their method is general and not optimized for LIT-SiGamal specifically. In this section, we modify their algorithm and introduce novel propositions for computing $(2,2)$-isogenies tailored to the LIT-SiGamal setting.

C.1 Introduction of theta coordinates

In this subsection, we introduce theta coordinates and their properties for computing isogenies. See [33] and [17] for more details.

Let k be a field, let A be an abelian variety over k of dimension g, and let $\mathcal{L}$ be a totally symmetric ample invertible sheaf of separable type over A. I.e., $\mathcal{L}$ is an ample invertible sheaf over A such that there is an isomorphism $\varphi\colon \mathcal{L} \xrightarrow{\sim} [-1]^*\mathcal{L}$ satisfying $\varphi_P = \mathrm{id}_{\mathcal{L}_P}$ for any $P \in A$ of order 2, and the dimension of the space of the global sections $\Gamma(A,\mathcal{L})$ is coprime to the characteristic of k. We define $H(\mathcal{L})$ as a subgroup $\{P \in A \mid T_P^*\mathcal{L} \cong \mathcal{L}\}$, where T_P is the translation-by-P map over A. Then, there is a sequence $D = (d_1, \ldots, d_g)$ with $d_{i+1} | d_i$ $(i = 1, \ldots, g-1)$ and $d_1 > 1$ such that

$$H(\mathcal{L}) \cong \bigoplus_{i=1}^{g} \mathbb{Z}/d_i\mathbb{Z} \oplus \mathrm{Hom}\left(\bigoplus_{i=1}^{g} \mathbb{Z}/d_i\mathbb{Z},\ \overline{k}^{\times}\right).$$

We say $\mathcal{L}$ is of type D. We denote $\oplus_{i=1}^{g}\mathbb{Z}/d_i\mathbb{Z}$ by $K(D)$ and $\mathrm{Hom}(K(D), \overline{k}^{\times})$ by $\hat{K}(D)$. We also define $\mathcal{G}(\mathcal{L})$ as

$$\mathcal{G}(\mathcal{L}) := \{(P, \phi_P) \mid x \in H(\mathcal{L}),\ \phi_P\colon T_P^*\mathcal{L} \xrightarrow{\sim} \mathcal{L}\}.$$

Then, the $\mathcal{G}(\mathcal{L})$ also has a group structure, and there is an exact sequence:

$$0 \longrightarrow \overline{k}^{\times} \longrightarrow \mathcal{G}(\mathcal{L}) \longrightarrow H(\mathcal{L}) \longrightarrow 0.$$

Define $\mathcal{G}(D) := \overline{k}^{\times} \oplus K(D) \oplus \hat{K}(D)$. Then, the above sequence is isomorphic to the following sequence:

$$0 \longrightarrow \overline{k}^{\times} \longrightarrow \mathcal{G}(D) \longrightarrow K(D) \oplus \hat{K}(D) \longrightarrow 0.$$

We call the isomorphism $\Theta_{\mathcal{L}} \colon \mathcal{G}(D) \xrightarrow{\sim} \mathcal{G}(\mathcal{L})$ a *theta structure of type* D. Let $V(D)$ be the space $\mathrm{Hom}(K(D), \overline{k})$. Then, the $V(D)$ is the unique irreducible representation of $\mathcal{G}(D)$. Similarly, the space of global sections $\Gamma(A, \mathcal{L})$ is the irreducible representation of $\mathcal{G}(\mathcal{L})$. Therefore, there is an isomorphism between $V(D)$ and $\Gamma(A, \mathcal{L})$ compatible with those group actions. The basis of $\Gamma(A, \mathcal{L})$ derived from the canonical basis of $V(D)$ (*i.e.,* the basis consisting of the Kronecher delta functions on $K(D)$) by the isomorphism is called *theta coordinates*. If $d_1 = \ldots = d_g = n$, we call them theta coordinates of *level* n.

Let $(\theta_{A,i})_{i=0,\ldots,2^g-1}$ be theta coordinates of level 2 over A. Let $S_1, \ldots, S_g$ be points of $A[2]$ derived from the canonical basis of $K(2, \ldots, 2)$, and let $T_1, \ldots, T_g$ be those from $\hat{K}(2, \ldots, 2)$. Theta coordinates $(\theta_{A,i})_i$ can be determined by a symplectic basis $(S'_1, \ldots, S'_g, T'_1, \ldots, T'_g)$ of $A[4]$ with $2S'_i = S_i$ and $2T'_i = T_i$ for $i = 0, \ldots, 2^g - 1$. By using theta coordinates, we can represent points in A by elements in $\mathbb{P}^{2^g-1}$ as follows:

$$P \longmapsto (\theta_{A,0}(P) : \theta_{A,1}(P) : \ldots : \theta_{A,2^g-1}(P)) \in \mathbb{P}^{2^g-1}.$$

This map is an embedding of a Kummer variety $A/\{\pm 1\}$ to $\mathbb{P}^{2^g-1}$. We call $(\theta_{A,i}(0))_i$ the *theta-null point*, which represents A. Let $\mathcal{H}$ be the Hadamard matrix of order 2^g. We call $\mathcal{H}((\theta_{A,i})_i)$ the *dual theta coordinates* of $(\theta_{A,i})_i$ and denote them by $(\tilde{\theta}_{A,i})_i$. The dual theta coordinates of $(\theta_{A,i})_i$ are also theta coordinates of A. It is easy to see that $(\tilde{\tilde{\theta}}_{A,i})_i = (\theta_{A,i})_i$. There is a symplectic basis $(\tilde{S}'_1, \ldots, \tilde{S}'_g, \tilde{T}'_1, \ldots, \tilde{T}'_g)$ associated with $(\tilde{\theta}_{A,i})_i$ such that

$$(\tilde{\theta}_{A,i}(\tilde{T}'_j))_i = \mathcal{H}((\theta_{A,i}(S'_j))_i), \quad (\tilde{\theta}_{A,i}(\tilde{S}'_j))_i = \mathcal{H}((\theta_{A,i}(T'_j))_i).$$

Define $K_1 := \langle S_1, \ldots, S_g \rangle$ and $K_2 := \langle T_1, \ldots, T_g \rangle$. Let f be a separable isogeny $f \colon A \to B$ with $\ker f = K_2$. The following properties of theta coordinates are important to construct algorithms for computing $(2,2)$-isogenies:

- $(\theta_{A,i}(P))_i = (\theta_{A,i}(-P))_i$,
- $(\theta_{A,i}(P + T_j))_i = ((-1)^{\langle i|j \rangle} \theta_{A,i}(P))_i$,
- $(\theta_{A,i}(P + S_j))_i = (\theta_{A,i+j}(P))_i$,
- There are theta coordinates $(\theta_{B,i})_i$ of B such that

  $$(\theta_{A,i}(P+Q) \cdot \theta_{A,i}(P-Q))_i = \mathcal{H}((\tilde{\theta}_{B,i}(f(P)) \cdot \tilde{\theta}_{B,i}(f(Q)))_i),$$

  and points derived from the canonical basis of $K(2, \ldots, 2)$ (resp. $\hat{K}(2, \ldots, 2)$) by the theta structure associated with $(\theta_{B,i})_i$ are $f(S_1), \ldots, f(S_g)$ (resp. $f(T'_1), \ldots, f(T'_g)$) (the duplication formula),
- $\theta_{A,i}(T'_j) = 0$ if $\langle i|j \rangle = 1$,
- $\theta_{A,i}(S'_j) = \theta_{A,i+j}(S'_j)$.

Algorithm 2 Change of a basis

Require: Montgomery curves E, E', a basis $\{P, Q\}$ of $E[4]$, and a basis $\{P', Q'\}$ of $E'[4]$ such that $x(P) = x(P') = 1$
Ensure: The basis-change matrix N
1: $H_1 \leftarrow \mathsf{action_by_translation}(Q)$ ([17, Algorithm 1])
2: $H_2 \leftarrow \mathsf{action_by_translation}(Q')$
3: $n_{00} \leftarrow h_{00|1} \cdot h_{00|2} + h_{10|1} \cdot h_{10|2} + 1$
4: $n_{01} \leftarrow h_{00|1} \cdot h_{10|2} + h_{10|1} \cdot h_{00|2}$
5: $n_{02} \leftarrow h_{10|1} \cdot h_{00|2} + h_{00|1} \cdot h_{10|2}$
6: $n_{03} \leftarrow n_{00}$
7: $n_{10} \leftarrow h_{00|2} \cdot n_{00} + h_{01|2} \cdot n_{01}$
8: $n_{11} \leftarrow h_{10|2} \cdot n_{00} + h_{11|2} \cdot n_{01}$
9: $n_{12} \leftarrow h_{00|2} \cdot n_{02} + h_{01|2} \cdot n_{03}$
10: $n_{13} \leftarrow h_{10|2} \cdot n_{02} + h_{11|2} \cdot n_{03}$
11: $\begin{pmatrix} n_{20} & n_{21} & n_{22} & n_{23} \end{pmatrix} \leftarrow \begin{pmatrix} n_{02} & n_{03} & n_{00} & n_{01} \end{pmatrix}$
12: $\begin{pmatrix} n_{30} & n_{31} & n_{32} & n_{33} \end{pmatrix} \leftarrow \begin{pmatrix} n_{12} & n_{13} & n_{10} & n_{11} \end{pmatrix}$
13: **return** N

Here, we assume that the indices i and j belong to $K(2, \dots, 2)$ by the bijection map

$$\begin{array}{ccc} K(2, \dots, 2) & \longrightarrow & \{0, \dots, 2^g - 1\} \\ (a_1, \dots, a_g) & \longmapsto & \sum_{l=1}^{g} a_l 2^{l-1} \end{array},$$

and $\langle \cdot \mid \cdot \rangle \colon K(2, \dots, 2)^2 \to \mathbb{Z}/2\mathbb{Z}$ is the inner product over $K(2, \dots, 2)$. By using the above properties, we can construct algorithms to compute $(2, 2)$-isogenies (see [17]).

C.2 Computing theta coordinates from a product of Montgomery curves

In the LIT-SiGamal setting, we compute an isogeny from a product of two Montgomery curves to an abelian variety of dimension 2. Therefore, in order to use the formulas of theta coordinates, we first determine a theta structure of a product of Montgomery curves. We find that [17, Algorithm 2] determines theta coordinates of a product of two Montgomery curves $E \times E'$ determined by $T_1' = (P, P')$, $T_2' = (Q, Q')$, $S_1' = (0, Q')$, and $S_2' = (P, 0)$, where $\{P, Q\}$ is a basis of $E[4]$ and $\{P', Q'\}$ is that of $E'[4]$. Precisely speaking, [17, Algorithm 2] outputs a basis-change matrix N and we can easily compute the theta coordinates of (R, R') by computing $N \cdot {}^t(x(R)x(R') : x(R) : x(R') : 1)$. We can assume that $x(P) = x(P') = 1$ in the LIT-SiGamal setting; therefore, we can simplify [17, Algorithm 2] for LIT-SiGamal. Algorithm 2 shows our new algorithm to obtain a basis-change matrix N from a product of two Montgomery curves.

C.3 Property of special codomain

Let $(\theta_{E \times E', i})_i$ be theta coordinates of $E \times E'$ obtained by the above method, let K_2 be a subgroup of $E \times E'$ generated by T_1 and T_2, and let $f \colon E \times E' \to B$ be a

$(2,2)$-isogeny with $\ker f = K_2$. Denote theta coordinates of B by $(\theta_{B,i})_i$. Then, it has been known that one of $\tilde{\theta}_{B,i}(0)$s may be zero. We prove in Proposition 5 that $\tilde{\theta}_{B,3}(0)$ is always 0 in our construction of the theta coordinates of $E \times E'$.

Proposition 5. *Let E and E' be elliptic curves, let $\{P, Q\}$ be a basis of $E[4]$, and let $\{P', Q'\}$ be a basis of $E'[4]$. Assume that $(\theta_{E\times E',i})_i$ are theta coordinates of $E \times E'$ determined by $T_1' = (P, P')$, $T_2' = (Q, Q')$, $S_1' = (0, Q')$, and $S_2' = (P, 0)$. Denote a subgroup $\langle 2T_1', 2T_2'\rangle$ by K_2, and $(E \times E')/K_2$ by B. Let $(\theta_{B,i})_{i=0,1,2,3}$ be theta coordinates of B satisfying the duplication formula. Then, we have $\tilde{\theta}_{B,3}(0) = 0$.*

Proof. Note that it follows from the construction of $(\theta_{E\times E',i})_i$ that

$$\theta_{E\times E',i}((R, R')) = \theta_{E\times E',i}((-R, R')) = \theta_{E\times E',i}((R, -R'))$$

for any $(R, R') \in E \times E'$. Since we have $P = -3P$, it holds that

$$(\theta_{E\times E',i}(T_1'))_i = (\theta_{E\times E',i}(T_1' + 2S_2'))_i.$$

Recall that it holds that

$$(\theta_{E\times E',i}(P + 2S_2'))_i = (\theta_{E\times E',i+2}(P))_i,$$
$$(\theta_{E\times E',i}(T_1'))_i = (* : 0 : * : 0).$$

Therefore, we have

$$(\theta_{E\times E',i}(T_1'))_i = (1 : 0 : \pm 1 : 0).$$

It also holds that

$$(\theta_{E\times E',i}(T_2'))_i = (\theta_{E\times E',i}(T_2' + 2S_1'))_i,$$
$$\theta_{E\times E',i}(T_2') = (* : * : 0 : 0).$$

Therefore, by the same discussion as above, we have

$$(\theta_{E\times E',i}(T_2'))_i = (1 : \pm 1 : 0 : 0).$$

Denote by f the $(2,2)$-isogeny $E \times E' \to B$. It follows from the duplication formula that

$$(\tilde{\theta}_{B,i}(f(T_1')) \cdot \tilde{\theta}_{B,i}(0))_i = \mathcal{H}((\theta_{E\times E',i}(T_1')^2)_i) = (1 : \pm 1 : 0 : 0),$$
$$(\tilde{\theta}_{B,i}(f(T_2')) \cdot \tilde{\theta}_{B,i}(0))_i = \mathcal{H}((\theta_{E\times E',i}(T_2')^2)_i) = (1 : 0 : \pm 1 : 0).$$

Denote $(\tilde{\theta}_{B,i}(0))_i$ by $(\alpha : \beta : \gamma : \delta)$. Note that $f(T_1')$ and $f(T_2')$ are points derived from the canonical basis of $K(2,2)$ by the theta structure related to $(\tilde{\theta}_{B,i})_i$. Therefore, it holds that

$$(\tilde{\theta}_{B,i}(f(T_1')))_i = (\beta : \alpha : \delta : \gamma),$$
$$(\tilde{\theta}_{B,i}(f(T_2')))_i = (\gamma : \delta : \alpha : \beta),$$

and we have $\beta\delta = 0$ and $\alpha\beta \neq 0$. Hence, we conclude $\delta = 0$. □

Algorithm 3 Projective inversion (proj_inv)

Require: $(a_0 : a_1 : \ldots : a_m) \in \mathbb{P}^m$
Ensure: $(a_0^{-1} : a_1^{-1} : \ldots : a_m^{-1}) \in \mathbb{P}^m$
if $m = 0$ **then**
  return (1)
else if $m = 1$ **then**
  return $(a_1 : a_0)$
else
  $(b_0 : \ldots : b_{\lfloor m/2 \rfloor}) \leftarrow \mathsf{proj_inv}((a_0 : \ldots : a_{\lfloor m/2 \rfloor}))$
  $(b_{\lfloor m/2 \rfloor + 1} : \ldots : b_m) \leftarrow \mathsf{proj_inv}((a_{\lfloor m/2 \rfloor + 1} : \ldots : a_m))$
  $a_{\mathrm{left}} \leftarrow b_0 \cdot a_0$
  $a_{\mathrm{right}} \leftarrow b_m \cdot a_m$
  for $i = 0, \ldots, \lfloor m/2 \rfloor$ **do**
    $b_i \leftarrow a_{\mathrm{right}} \cdot b_i$
  end for
  for $i = \lfloor m/2 \rfloor + 1, \ldots, m$ **do**
    $b_i \leftarrow a_{\mathrm{left}} \cdot b_i$
  end for
  return $(b_0 : b_1 : \ldots : b_m)$
end if

C.4 Computation of $(2, 2)$-isogenies using the projective inversion

In [17], the authors proposed the algorithms using the batched inversion method for computing inversions of multiple elements in $\mathbb{F}_{p^2}$. Since theta coordinates embed points to the projective space $\mathbb{P}^3$, we do not need to compute the exact inversions of the given elements in some cases. Therefore, we can reduce the computational cost of some of the algorithms for theta coordinates by using the projective inversion (Algorithm 3) instead of the batched inversion.

Algorithms 4 and 5 are algorithms to compute the codomain variety of a $(2, 2)$-isogeny using the projective inversion. These are respectively more efficient than algorithms proposed in [17] (*i.e.,* [17, Algorithm 6] and [17, Algorithm 7]).

Remark 11. The outputs of [17, Algorithms 6 and 7] are supposed to have $\alpha = 1$, while those of our novel algorithms are not. Therefore, there is a possibility that the total computational cost to compute $(2, 2)$-isogenies using Algorithms 4 and 5 is less efficient than the original algorithm in [17] because extra computations using α occur in the total computation. In particular, m extra multiplications occur in evaluating m points.

In our implementation, the total algorithm becomes more efficient by replacing all [17, Algorithms 6 and 7] to Algorithms 4 and 5. However, for the above reason, it is thought that the most efficient algorithm for computing $(2, 2)$-isogenies uses both [17, Algorithms 6 and 7] and Algorithms 4 and 5. The optimization is our future work.

Algorithm 4 Codomain

Require: Theta coordinates $(x_{T_1''} : y_{T_1''} : z_{T_1''} : w_{T_1''})$ and $(x_{T_2''} : y_{T_2''} : z_{T_2''} : w_{T_2''})$ of eight torsion points T_1'' and T_2'' such that $K_2 = \langle 2T_1'', 2T_2'' \rangle$
Ensure: The dual theta-null point $(\alpha : \beta : \gamma : \delta)$ of A/K_2 and $(\alpha^{-1} : \beta^{-1} : \gamma^{-1} : \delta^{-1})$ and the theta-null point $(a : b : c : d)$ of A/K_2

$(x_1 : y_1 : z_1 : w_1) \leftarrow \mathcal{H} \circ \mathcal{S}(x_{T_1''} : y_{T_1''} : z_{T_1''} : w_{T_1''})$
$(x_2 : y_2 : z_2 : w_2) \leftarrow \mathcal{H} \circ \mathcal{S}(x_{T_2''} : y_{T_2''} : z_{T_2''} : w_{T_2''})$
$(\alpha_0 : \gamma_0 : \delta_0) \leftarrow \mathsf{proj_inv}((x_1 : x_2 : y_2))$
$\alpha \leftarrow x_1 \cdot \alpha_0$
$\beta \leftarrow y_1 \cdot \alpha_0$
$\gamma \leftarrow z_2 \cdot \gamma_0 \cdot \alpha$
$\delta \leftarrow w_2 \cdot \delta_0 \cdot \beta$
$\beta \leftarrow \beta \cdot \alpha$
$\alpha \leftarrow \alpha^2$
$(a : b : c : d) \leftarrow \mathcal{H}(\alpha : \beta : \gamma : \delta)$
return $(\alpha : \beta : \gamma : \delta)$, $\mathsf{proj_inv}((\alpha : \beta : \gamma : \delta))$, $(a : b : c : d)$

Algorithm 5 Codomain if $\delta = 0$

Require: Theta coordinates $(x_{T_1''} : y_{T_1''} : z_{T_1''} : w_{T_1''})$ and $(x_{T_2''} : y_{T_2''} : z_{T_2''} : w_{T_2''})$ of eight torsion points T_1'' and T_2'' such that $K_2 = \langle 2T_1'', 2T_2'' \rangle$
Ensure: The dual theta-null point $(\alpha : \beta : \gamma : 0)$ of A/K_2 and $(\alpha^{-1} : \beta^{-1} : \gamma^{-1} : 0)$ and the theta-null point $(a : b : c : d)$ of A/K_2

$(x_1 : y_1 : z_1 : w_1) \leftarrow \mathcal{H} \circ \mathcal{S}(x_{T_1''} : y_{T_1''} : z_{T_1''} : w_{T_1''})$
$(x_2 : y_2 : z_2 : w_2) \leftarrow \mathcal{H} \circ \mathcal{S}(x_{T_2''} : y_{T_2''} : z_{T_2''} : w_{T_2''})$
$(\alpha_0 : \gamma_0) \leftarrow \mathsf{proj_inv}((x_1 : x_2))$
$\alpha \leftarrow x_1 \cdot \alpha_0$
$\beta \leftarrow y_1 \cdot \alpha_0$
$\gamma \leftarrow z_2 \cdot \gamma_0$
$(a : b : c : d) \leftarrow \mathcal{H}(\alpha : \beta : \gamma : 0)$
return $(\alpha : \beta : \gamma : 0)$, $(\mathsf{proj_inv}((\alpha : \beta : \gamma)) : 0)$, $(a : b : c : d)$

C.5 Computation of an isogenies between products of two elliptic curves

Let E, E', E_1, E_1' be elliptic curves, and let $\Phi \colon E \times E' \to E_1 \times E_1'$ be a $(2^a, 2^a)$-isogeny with $\ker \Phi = \langle (P, P'), (Q, Q') \rangle$, where $\{P, Q\}$ is a basis of $E[2^a]$ and $\{P', Q'\}$ is that of $E'[2^a]$. In the LIT-SiGamal setting, we compute this isogeny by computing two isogenies $\Phi_0 \colon E \times E' \to V$ and $\Phi_1 \colon E_1 \times E_1' \to V$ such that $\Phi = \hat{\Phi}_1 \circ \Phi_0$. However, theta coordinates of V are not unique, and the representation of the codomain of Φ_0 and that of Φ_1 are not the same if we use the method proposed in [17]. Therefore, to compute Φ, we need to detect the isomorphism between two representations.

We prove in Proposition 6 that the following matrix gives the isomorphism between two representations in our construction:

$$\mathcal{H} \cdot \begin{pmatrix} 1\,0\,0\,0 \\ 0\,0\,1\,0 \\ 0\,1\,0\,0 \\ 0\,0\,0\,1 \end{pmatrix}.$$

By using this transformation, we can evaluate image points under $\hat{\Phi}_1 \circ \Phi_0$.

Proposition 6. *Let E, E', E_1, E_1' be elliptic curves. Assume that there is an isogeny diamond as follows:*

$$\begin{array}{ccc} E_1 & \xrightarrow{\phi} & E \\ {\scriptstyle \phi_1}\downarrow & & \downarrow{\scriptstyle \phi_1'} \\ E' & \xrightarrow[\phi']{} & E_1' \end{array}$$

Here, we also assume that $\deg\phi + \deg\phi_1 = 2^a$ and $\deg\phi_1 \equiv -1 \pmod 4$ for an integer $a \geq 2$. Let $\Phi\colon E \times E' \to E_1 \times E_1'$ be a $(2^a, 2^a)$-isogeny represented by

$$\Phi = \begin{pmatrix} \hat{\phi} & \hat{\phi_1} \\ -\phi_1' & \phi' \end{pmatrix}.$$

Let a_1, a_2 be integers with $a = a_1 + a_2$, let $\{P, Q\}$ be a basis of $E_1[2^a]$, let Φ_0 be a separable isogeny from $E \times E'$ with

$$\ker\Phi_0 = \langle (2^{a_2}\phi(P), 2^{a_2}\phi_1(P)), (2^{a_2}\phi(Q), 2^{a_2}\phi_1(Q)) \rangle,$$

and let Φ_1 be a separable isogeny from $E_1 \times E_1'$ with

$$\ker\Phi_1 = \langle (2^{a_1}(\deg\phi_1)P, 2^{a_1}(\phi_1' \circ \phi)(P)), (2^{a_1}(\deg\phi_1)Q, 2^{a_1}(\phi_1' \circ \phi)(Q)) \rangle.$$

Note that $\Phi = \hat{\Phi}_1 \circ \Phi_0$. Denote by V the codomain abelian variety of Φ_0 and Φ_1.

Let $(\theta_{V,i})_i$ be theta coordinates of V determined by the symplectic basis

$$\begin{aligned} &(\Phi_0((0, 2^{a-2}\phi_1(Q))),\ \Phi_0((2^{a-2}\phi(P), 0)), \\ &\quad \Phi_0((2^{a_2-2}\phi(P), 2^{a_2-2}\phi_1(P))),\ \Phi_0((2^{a_2-2}\phi(Q), 2^{a_2-2}\phi_1(Q)))), \end{aligned}$$

and let $(\theta_{V,i}')_i$ be theta coordinates of V determined by the symplectic basis

$$\begin{aligned} &(\Phi_1((0, 2^{a-2}(\phi_1' \circ \phi)(Q))),\ \Phi_1((2^{a-2}P, 0)), \\ &\quad \Phi_1((2^{a_2-2}P, -2^{a_2-2}(\phi_1' \circ \phi)(P))),\ \Phi_1((-2^{a_2-2}Q, 2^{a_2-2}(\phi_1' \circ \phi)(Q)))). \end{aligned}$$

Then, for any $R \in E \times E'$, it holds that

$${}^t(\theta_{V,i}'(\Phi_0(R)))_i = \mathcal{H} \cdot \begin{pmatrix} 1\,0\,0\,0 \\ 0\,0\,1\,0 \\ 0\,1\,0\,0 \\ 0\,0\,0\,1 \end{pmatrix} \cdot {}^t(\theta_{V,i}(\Phi_0(R)))_i.$$

Proof. It suffices to find the isomorphism mapping the symplectic base associated with $(\theta_{V,i})_i$ to that associated with $(\theta'_{V,i})_i$. We denote the basis associated with $(\theta_{V,i})_i$ by

$$(S_1, S_2, T_1, T_2)$$

and that associated with $(\theta'_{V,i})_i$ by

$$(S'_1, S'_2, T'_1, T'_2).$$

It is clear that $S_1 = 2^{a-2}\Phi_0((0, \phi_1(Q)))$, and it follows from $\deg \phi_1 \equiv -1 \pmod 4$ that

$$T'_2 = 2^{a_1-2}(\Phi_1 \circ \Phi)((0, \phi_1(Q))) = 2^{a-2}\Phi_0((0, \phi_1(Q))).$$

Therefore, it holds that $S_1 = T'_2$. Similarly, we have $S_2 = T'_1$. Note that $\hat{\Phi}$ can be represented by

$$\hat{\Phi} = \begin{pmatrix} \phi & -\hat{\phi}'_1 \\ \phi_1 & \hat{\phi}' \end{pmatrix}.$$

We have

$$T_2 = 2^{a_2-2}(\Phi_0 \circ \hat{\Phi})((0, (\phi'_1 \circ \phi)(Q))) = 2^{a-2}\Phi_1((0, (\phi'_1 \circ \phi)(Q))) = S'_1.$$

Similarly, we also have $T_1 = S'_2$. Therefore, the isomorphism mapping (S_1, S_2, T_1, T_2) to (S'_1, S'_2, T'_1, T'_2) is given by the composition of the following two maps:

- The isomorphism mapping (S_1, S_2, T_1, T_2) to (S_2, S_1, T_2, T_1).
- The isomorphism mapping (S_1, S_2, T_1, T_2) to (T_1, T_2, S_1, S_2),

The first isomorphism is represented by the matrix

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

and the second one is represented by $\mathcal{H}$. This completes the proof of the proposition. □

C.6 Computation of a product of Montgomery curves from theta coordinates

At the end of the computation of a $(2^a, 2^a)$-isogeny, we transform a point given by theta coordinates to a point in a product of two Montgomery curves. The basic method to obtain the representation of a point in Montgomery curves from theta coordinates is found in [17, Section 4.1], and we can simplify this method using Proposition 5. However, this method may not be directly applicable to LIT-SiGamal. This is because, in the LIT-SiGamal setting, we want to obtain a representation in a fixed product of Montgomery curves $E \times E'$, while the basic

method may provide a representation in $E_1 \times E_1'$, which is isomorphic to, but not the same as, $E \times E'$. Therefore, we need to modify the method for LIT-SiGamal.

Let E and E' be Montgomery curves, and $(\theta_{E\times E',i})_i$ be theta coordinates determined by

$$((0,Q'),(P,0),(P,P'),(Q,Q')),$$

where $\{P,Q\}$ and $\{P',Q'\}$ are bases of $E[4]$ and $E'[4]$ respectively such that $x(P) = x(P') = 1$. The aim is to find the method to obtain $x(R)$ and $x(R')$ from $(\theta_{E\times E',i}((R,R')))_i$. Denote $\mathcal{H}((\theta_{E\times E',i}((R,R')))_i)$ by $(a:b:c:d)$. From the method in [17, Section 4.1] and Proposition 5, we can represent (R,R') by the pair $((a:\sqrt{-1}b),(a:\sqrt{-1}c))$. We note that $((a:\sqrt{-1}b),(a:\sqrt{-1}c))$ is $((\theta_{E,i}(R))_i,(\theta_{E',i}(R'))_i)$ or $((\theta_{E',i}(R'))_i,(\theta_{E,i}(R))_i)$, where $(\theta_{E,i})_i$ (resp. $(\theta_{E',i})_i$) are theta coordinates of E (resp. E') defined by an appropriate theta structure. We now discuss the way to compute $x(R)$ and $x(R')$ from $(a:\sqrt{-1}b)$ and $(a:\sqrt{-1}c)$. We first focus on $(2P,0)$ and $(P,0)$. Denote the dual theta-null point of $E\times E'$ by $(\alpha:\beta:\gamma:\delta)$. We have

$$\begin{aligned}(\alpha:\beta:-\gamma:-\delta) &= \mathcal{H}((\theta_{E\times E',i}((2P,0)))_i),\\ (*:*:0:0) &= \mathcal{H}((\theta_{E\times E',i}((P,0)))_i),\\ (\alpha:\beta:\gamma:\delta) &= \mathcal{H}((\theta_{E\times E',i}((0,0)))_i).\end{aligned}$$

Therefore, it holds that

$$\begin{aligned}\{(\theta_{E,i}(2P))_i,(\theta_{E',i}(0))_i\} &= \{(\alpha:\sqrt{-1}\beta),(\alpha:-\sqrt{-1}\gamma)\},\\ \{(\theta_{E,i}(0))_i,(\theta_{E',i}(0))_i\} &= \{(\alpha:\sqrt{-1}\beta),(\alpha:\sqrt{-1}\gamma)\},\\ \{(\theta_{E,i}(P))_i,(\theta_{E',i}(0))_i\} &\supset \{(1:0)\}.\end{aligned}$$

Note that it follows from $\pm 2P \neq 0$ that $(\theta_{E,i}(2P))_i \neq (\theta_{E,i}(0))_i$. We conclude that

$$\begin{aligned}(\theta_{E',i}(0))_i = (\alpha:\sqrt{-1}\beta), \quad (\theta_{E,i}(2P))_i = (\alpha:-\sqrt{-1}\gamma),\\ (\theta_{E,i}(P))_i = (1:0), \quad (\theta_{E,i}(0))_i = (\alpha:\sqrt{-1}\gamma).\end{aligned}$$

From the above observation, we also have

$$(\theta_{E',i}(R'))_i = (a:\sqrt{-1}b), \quad (\theta_{E,i}(R))_i = (a:\sqrt{-1}c).$$

Recall that $x(P) = 1$ and $x(2P) = 0$. The transformation from $(\theta_{E,i}(R))_i$ to $x(R)$ is given a 2×2-matrix M_T. This matrix needs to satisfy

$$\begin{pmatrix}*\\0\end{pmatrix} = M_T\begin{pmatrix}\alpha\\\sqrt{-1}\gamma\end{pmatrix}, \quad \begin{pmatrix}0*\end{pmatrix} = M_T\begin{pmatrix}\alpha\\-\sqrt{-1}\gamma\end{pmatrix}, \quad *\begin{pmatrix}1\\1\end{pmatrix} = M_T\begin{pmatrix}1\\0\end{pmatrix}.$$

Therefore, we have

$$M_T = \begin{pmatrix}\sqrt{-1}\gamma & \alpha\\ \sqrt{-1}\gamma & -\alpha\end{pmatrix}.$$

Algorithm 6 Theta coordinates to Montgomery curves

Require: Theta coordinates $(\theta_{E\times E',i}((R,R')))_i$ and theta-null point $(\theta_{E\times E',i}((0,0)))_i$ of $E \times E'$
Ensure: $x(R)$ and $x(R')$
1: $(a : b : c : d) \leftarrow \mathcal{H}((\theta_{E\times E',i}((R,R')))_i)$
2: $(X_R : Z_R) \leftarrow (a : \sqrt{-1}c)$
3: $(X_{R'} : Z_{R'}) \leftarrow (a : \sqrt{-1}b)$
4: $(\alpha : \beta : \gamma : \delta) \leftarrow \mathcal{H}((\theta_{E\times E',i}((0,0)))_i)$
5: $M_T \leftarrow \begin{pmatrix} \sqrt{-1}\gamma & \alpha \\ \sqrt{-1}\gamma & -\alpha \end{pmatrix}$
6: $\begin{pmatrix} X_R \\ Z_R \end{pmatrix} \leftarrow M_T \cdot \begin{pmatrix} X_R \\ Z_R \end{pmatrix}$
7: $M'_T \leftarrow \begin{pmatrix} -\sqrt{-1}\alpha & \beta \\ \sqrt{-1}\beta & -\alpha \end{pmatrix}$
8: $\begin{pmatrix} X_{R'} \\ Z_{R'} \end{pmatrix} \leftarrow M'_T \cdot \begin{pmatrix} X_{R'} \\ Z_{R'} \end{pmatrix}$
9: **return** $(X_R : Z_R)$ and $(X_{R'} : Z_{R'})$

We next focus on $(2P, 2P')$ and (P, P'). From a similar discussion as above, we have

$$(\theta_{E',i}(2P'))_i = (\beta : \sqrt{-1}\alpha), \quad (\theta_{E',i}(P'))_i = (1 : \sqrt{-1}).$$

Therefore, from $x(2P') = 0$ and $x(P') = 1$, the representation matrix M'_T of the transformation from $(\theta_{E',i}(R'))_i$ to $x(R')$ satisfies

$$\begin{pmatrix} * \\ 0 \end{pmatrix} = M'_T \begin{pmatrix} \alpha \\ \sqrt{-1}\beta \end{pmatrix}, \quad \begin{pmatrix} 0 \\ * \end{pmatrix} = M'_T \begin{pmatrix} \beta \\ \sqrt{-1}\alpha \end{pmatrix}, \quad * \begin{pmatrix} 1 \\ 1 \end{pmatrix} = M'_T \begin{pmatrix} 1 \\ \sqrt{-1} \end{pmatrix}.$$

We have

$$M'_T = \begin{pmatrix} -\sqrt{-1}\alpha & \beta \\ \sqrt{-1}\beta & -\alpha \end{pmatrix}.$$

From the above discussion, we can construct an algorithm to compute $x(R)$ and $x(R')$ from $(\theta_{E\times E',i}((R,R')))_i$ (Algorithm 6).