Transforming the POKE public key Protocol into a Key Encapsulation Mechanism.

Nouhou Abdou Idris[1], Yunusa Abdulsalam[2], and Mustapha Hedabou[3]

[1] Mohammed VI polytechnic University, Benguerir, Morocco
[2] Abdou.idris-NOUHOU@um6p.ma
[3] Abdulsalam.YUNUSA@um6p.ma
Mustapha.HEDABOU@um6p.ma

Abstract. The inherent presence of large-scale quantum computers is deemed to threaten the traditional public-key cryptographic systems. As a result, NIST has necessitated the development of post-quantum cryptography (PQC) algorithms. Isogeny-based schemes have compact key sizes; however, there have been some fundamental vulnerabilities in the smooth-degree isogeny systems. Therefore, the literature has concentrated its efforts on higher-dimensional isogenies of non-smooth degrees. In this work, we present POKE-KEM, a key encapsulation mechanism (KEM) derived from the POKE public-key encryption (PKE) scheme via a Fujisaki-Okamoto (FO) transformation with its corresponding optimized security levels during transformation. Despite POKE's efficiency advantages, its current formulation provides only IND-CPA security, limiting practical deployment where adaptive chosen-ciphertext security is essential. The resulting POKE-KEM construction achieves IND-CCA security under the computational hardness of the C-POKE problem in both the random oracle model (ROM) and quantum random oracle model (QROM). Our implementation demonstrates significant practical advantages across all NIST security levels (128, 192, and 256 bits), supporting the practical viability of POKE-KEM for post-quantum cryptographic deployments. We provide a comprehensive security analysis including formal proofs of correctness, rigidity, and IND-CCA security. Our underlying construction leverages the difficulty of recovering isogenies with unknown, non-smooth degrees, resistant to known quantum attacks.

Keywords: Post-quantum cryptography · Isogeny-based cryptography · Key encapsulation mechanism · Fujisaki-Okamoto transform · IND-CCA security · Higher-dimensional isogenies · C-POKE problem

1 Introduction

The advent of large-scale quantum computers poses a significant threat to our current cryptographic infrastructure, which relies on mathematical problems that are computationally hard for classical computers, such as the integer factorization problem and the discrete logarithm problem [34],[35],[36],[37],[38]].

These problems can be solved efficiently by quantum algorithms like Shor's algorithm [35], rendering widely used public-key cryptographic schemes like RSA and Elliptic Curve Cryptography (ECC) insecure. This imminent threat has necessitated the development of post-quantum cryptography (PQC), which aims to create cryptographic systems that are secure against both classical and quantum adversaries. Since the NIST announcement on the depreciation of weak traditional cryptographic algorithms by 2030 [51], researchers have explored several promising mathematical families for building PQC schemes with different levels of hardness against quantum threats. The families include: Lattice-based cryptography, relies on the hardness of problems involving lattices; Hash-based cryptography uses cryptographic hash functions to construct digital signatures, offering excellent security; Code-based cryptography, based on the difficulty of decoding linear codes; Isogeny-based cryptography, leverages the mathematical properties of isogenies between elliptic curves

Though the Isongeny-based approach is known for its elegant mathematical structure and relatively compact key sizes, there have been some polynomial-time attacks [13] against Supersingular Isogeny Diffie-Hellman (SIDH) [14] that exploit the fundamental structural weakness in smooth-degree isogeny systems, particularly the auxiliary torsion point information that enabled efficient isogeny evaluation but inadvertently revealed secret pathways. Following the devastating attack, the literature has now seen a transition from one-dimensional elliptic curves to higher-dimensional abelian varieties that represent a fundamental paradigm shift in isogeny-based cryptography. Most notable approach is using mathematical structures such as the two-dimensional abelian surfaces and Kani's lemma [30], providing algebraic flexibility while maintaining computational tractability. Some protocols in the literature have also exemplified a similar approach, including SQIsignHD [3], which extends quaternion-based signatures to higher dimensions; FESTA [6], designed explicitly to resist SIDH-style attacks; and QFESTA [7], which optimizes higher-dimensional computations through theta coordinate representations. These protocols operate not on elliptic curves (dimension one) but on higher-dimensional abelian varieties, such as abelian surfaces. While these schemes demonstrate the viability of higher-dimensional isogeny cryptography, they face significant challenges, including computational overhead, large ciphertext sizes, and complex parameter selection.

1.1 Motivation

Among recent higher-dimensional developments, the POKE [9], a public key encryption from a higher-dimensional Isogenies protocol, stands out due to its efficiency-security trade-off. The protocol leverages isogenies of unknown, non-smooth degree, rather than relying on smooth-degree isogenies like its predecessors. Their design approach addresses key vulnerabilities that enabled attacks on SIDH while maintaining computational efficiency through specialized evaluation algorithms. The security of POKE rests on the Computational POKE (C-POKE) assumption, which posits the computational hardness of recovering secret isogenies or evaluating them at arbitrary points given only public information. This

assumption appears resistant to known SIDH attacks, due to fundamental differences in the mathematical structure and the use of non-smooth degrees.

Despite POKE's impressive efficiency characteristics, its current formulation provides only indistinguishability under chosen-plaintext attacks (IND-CPA) security. Modern cryptographic applications, particularly in post-quantum contexts, require the stronger notion of indistinguishability under adaptive chosen-ciphertext attacks (IND-CCA) security, in addition to tightly reducing the underlying C-POKE assumption, avoiding excessive security loss that would require impractically large parameters. These security levels are essential for real-world deployment, where adversaries may have access to decryption oracles or partial information about encrypted data.

The gap between POKE's computational efficiency and the practical security requirements for deployment creates a critical challenge. Simply applying standard KEM transforms may compromise POKE's performance advantages or introduce subtle security vulnerabilities. The transformation of semantically secure public-key encryption schemes into IND-CCA secure KEMs is well-studied through various versions of the Fujisaki-Okamoto (FO) transform [21]. However, applying these transforms to specific algebraic structures requires careful analysis to ensure security preservation while maintaining computational efficiency. The unique mathematical properties of higher-dimensional isogeny schemes necessitate specialized treatment in KEM construction.

1.2 Our contribution

This work presents POKE-KEM, a highly efficient and compact key encapsulation mechanism that leverages a Fujisaki-Okamoto (FO) style transformation, achieving IND-CCA security in both the random oracle model (ROM) and the quantum random oracle model (QROM) under the computational hardness of the C-POKE problem. This work builds upon the foundational POKE PKE protocol with the primary aim of achieving enhanced security guarantees. This transformation is crucial for practical applications, as KEMs are the preferred building block for hybrid encryption schemes in a post-quantum world. Our proposed construction offers several key advantages as follows:

- Improved Efficiency: POKE-KEM reduces ciphertext sizes by 30–50% compared to state-of-the-art schemes like QFESTA while maintaining competitive computational performance.
- Provable Security: We provide a tight reduction from the C-POKE problem to the OW-CPA and IND-CCA security of POKE-KEM, ensuring robustness against quantum adversaries.
- Improved Security: We provide adequate parameter optimizations in the key generation algorithm that strengthen the protocol's security.

Therefore, this work addresses the critical gap between POKE's efficiency and practical security requirements by presenting the first key encapsulation mechanism derived from the POKE protocol. Our specific contributions are as follows:

1. A formalization of the C-POKE one-wayness assumption under the mathematical formalism.
2. An FO-style transformation tailored to POKE, enabling IND-CCA security without costly re-encryption checks.
3. An implementation of POKE-KEM for three security levels (128, 192, and 256 bits), demonstrating 5–15× faster encapsulation.

1.3 Organization of the Paper

2 Related works

The mathematical foundation for higher-dimensional isogeny cryptography builds on Kani's lemma [30], which enables the construction of product isogenies between products of elliptic curves. The SQIsign signature scheme [10] and its higher-dimensional variant SQIsignHD [3] represent one branch of post-SIDH isogeny cryptography. These protocols leverage the Deuring correspondence between supersingular elliptic curves and quaternion algebras, achieving security through the difficulty of solving quaternion ideal problems. While primarily focused on signatures, SQIsignHD demonstrates the viability of higher-dimensional isogeny computations.

In terms of PKE and KEM variants, current post-quantum KEM candidates span a wide range of performance. Lattice-based KEMs like Kyber [2] offer excellent performance (sub-millisecond operations) but with larger key sizes (800-1500 bytes). Code-based KEMs ([1]) provide strong security foundations but suffer from very large key sizes (several KB). Isogeny-based approaches historically offered the most compact keys but faced SIDH attack challenges. The Fast Encryption from Supersingular Torsion Attacks (FESTA) protocol [6] was explicitly designed to resist SIDH-style attacks by carefully managing torsion point information. FESTA operated on products of elliptic curves and achieved IND-CCA security through a variant of the Fujisaki-Okamoto transform. However, FESTA suffers from large ciphertext sizes (approximately 400-500 bytes at 128-bit security) and computational overhead in higher-dimensional isogeny computations. Building on FESTA's foundation, QFESTA [7] was introduced to provide significant optimizations through theta coordinate representations and improved algorithms for dimension-two isogenies. QFESTA achieved better performance than FESTA but still faces challenges with ciphertext sizes (approximately 512 bytes at 128-bit security) and computational complexity. The protocol provides both PKE and KEM variants, with the KEM achieving IND-CCA security through careful parameter choices and transform design.

Basso and Maino designed the POKE protocol [9], a PKE from Higher-Dimensional Isogenies. They introduced a fundamentally different approach by leveraging isogenies of unknown, non-smooth degree rather than the smooth degrees used in SIDH and its successors. This design choice addresses a key vulnerability exploited in SIDH attacks while maintaining computational efficiency through specialized algorithms for non-smooth isogeny evaluation. POKE demonstrated significant performance improvements over previous schemes of

higher dimension through several innovations: optimized Kummer line arithmetic for x-only coordinate computations, efficient evaluation algorithms for non-smooth degree isogenies, and careful parameter selection balancing security and performance. These optimizations result in substantially faster key generation and encryption compared to FESTA [6] and QFESTA [7].

Our contribution of POKE-KEM provides the first IND-CCA secure KEM based on the efficient POKE protocol. This work combines the computational advantages of POKE's non-smooth degree approach with the security guarantees required for practical deployment. The resulting KEM offers a compelling point in the performance-security trade-off space, providing significantly better efficiency than existing higher-dimensional alternatives while maintaining strong security foundations.

3 Preliminaries

Isogeny-based cryptography was initially based on building secure cryptographic protocols using isogenies (isogenies of elliptic curves). Nowadays, isogenies-based cryptography relies on higher-dimensional isogenies that are proving to be very secure. In this section, we provide algebraic structures that are used in the protocol conception and further outline the mathematical computation problem that ensures the security of the protocol.

Definition 1 ([26],[32] ,[28]). *An elliptic curve on the Short Weierstrass form over a field k is the set of point $\{(x, y)\}$ verifying: $E : y^2 = x^3 + ax + b$ with $a, b \in k$*

Definition 2 ([26],[32]). *A montgomery curve over a field k is the set of point $\{(x, y)\}$ verifying: $E : by^2 = x(x^2 + ax + 1)$ with $a, b \in k$.*

Definition 3 ([26],[32]). *Let be $E : by^2 = x(x^2 + ax + 1)$ be a montgomery curve, a Kummer line is defined as $\mathcal{K} = E/\{\pm 1\}$.*

Remark 1 ([27]).

- Every given elliptic curve E can be written in short Weierstrass form. But depending on the property of E and the field k it can be put into different models as Montgomery, Theta, Twisted Edwards... For each model Montgomery, Theta and other, one considers their corresponding arithmetic for differential addition, doubling.
  Hold that the implementation of an isogenies-based protocol one uses a combination of different models for a shake of performance.
- The Kummer line gives tha advantage of working just on $x-$ coordinate. This is done for a shake a efficiency and sizes optimization.

Definition 4 ([32],[28],[27]). *An abelian variety A is a projective variety with a group structure. Furthermore, a Kummer variety is a variety defined by $\mathcal{K}_A = A/\pm 1$.*

Definition 5 ([32],[28],[27]). *An higher-dimensional isogeny is a special map between abelian varieties of dimension greater than one. These maps preserve the group structure of abelian varieties. Hold also that a Kummer isogeny is a map between Kummer varieties.*

The abelian varieties that are commonly used are lines and surfaces. So in most cases, we consider abelian lines and abelian surfaces. The higher-dimensional isogenies are built using abelian surfaces.

Remark 2 ([29]). Abelian surfaces are classified into the following two types:

- A product of elliptic curves $E_1 \times E_2$.
- The jacobian $Jac(\mathcal{C})$.

By corollary to remark 1, there exist four types of higher-dimensional isogenies of surfaces, namely:

- Splitting isogenies: $\phi : Jac(\mathcal{C}) \to E_1 \times E_2$
- Gluing isogenies: $\phi : E_1 \times E_2 \to Jac(\mathcal{C})$
- Product isogenies:$\phi : E_1 \times E_2 \to E_3 \times E_4$
- Generic isogenies: $\phi : Jac(\mathcal{C}) \to Jac(\mathcal{C})$

For our implementation, and what is also common in the literature, are the product isogenies, built based on Kani's lemma, which we recall in the next section.

Kani's lemma

Theorem 1. ***Kani's Lemma [30]*** *Let $f = f_1' \circ f_1 = f_2' \circ f_2$ be a (d_1, d_2)-isogeny diamond as above. Then*

$$F = \begin{pmatrix} f_1 & -\widetilde{f_1'} \\ f_2 & \widetilde{f_2'} \end{pmatrix}$$

is a d-isogeny $E_1 \times E_2 \to E_3 \times E_4$, where $d = d_1 + d_2$.

Furthermore, if f is an orthogonal isogeny diamond, then

$$\ker F = \{(d_1 x, -f(x)) \mid x \in A[d]\}.$$

3.1 Computational problems and hard problems

This subsection presents some computational problems due to isogenies. The main concept that abstracts the most in computational problems concerning isogenies is known as the Deuring correspondence. The Deuring correspondence is a correspondence between the category of supersingular elliptic curves and the category of the quaternion order. Table **??** presents some objects associations.[[6],[9],[5],[4],[3],[14][10]]

Elliptic Curve Side	**Quaternion Order Side (Deuring Correspondence)**
Supersingular elliptic curve $E/\mathbb{F}_{p^2}$	Quaternion order $\mathcal{O} \subseteq B_{p,\infty}$
Endomorphism ring $\mathrm{End}(E)$	Maximal order $\mathcal{O}$ in quaternion algebra $B_{p,\infty}$
Isogeny $\varphi : E \to E'$	Left ideal $I \subset \mathcal{O}$
Dual isogeny $\hat{\varphi} : E' \to E$	Right ideal $\mathcal{O}' = \mathrm{End}(E') = \mathrm{End}(I^{-1})$
Composition of isogenies $\varphi_2 \circ \varphi_1$	Ideal multiplication $I_1 I_2$
Degree of isogeny $\deg(\varphi)$	Norm of ideal $nrd(I)$
Isomorphism class of E	Isomorphism class of the order $\mathcal{O}$

Table 1: Deuring correspondence between supersingular elliptic curves and quaternion algebras

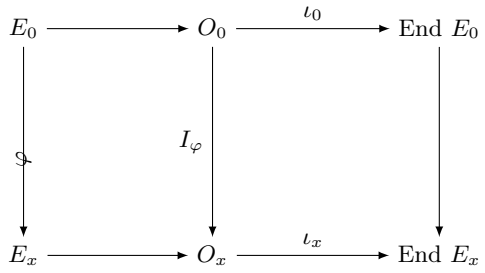


Fig. 1: Deuring correspondence between supersingular elliptic curves and quaternion algebras [8]

In getting an association between objects, we present below some well-known algorithms used in practice.

- **KernelToIsogeny [9],[7],[5],[5]:** Takes as input a supersingular elliptic curve E and subset $K \subset E[d]$. Then return an isogeny whose kernel is K.
- **CanonicalTorsionBasis [13]:** Takes as input E a supersingular elliptic curve and N an integer verifying $N/(p^2-1)$ and outputs the points P, Q such that $\langle P, Q \rangle = E[N]$.
- **EvalTorsion [13]** As input it takes a basis (P_1, P_2) of $E[D]$, an ideal I connecting $\mathcal{O} \cong \mathrm{End}(E)$ and $\mathcal{O}' := \mathrm{End}(E')$, two isogenies $\rho_1 : E_0 \to E$ and $\rho_2 : E_0 \to E'$ of respective degrees d_1 and d_2 prime to D and their respective kernel ideals I_1 and I_2.
  Then outputs $(\phi_I(P_1), \phi_I(P_2))$, where $\phi_I : E \to E'$ is the isogeny associated to I.

- **FullRepresentInteger [14]:** Takes as input a number $N > 4P$ and outputs $\gamma \in \mathcal{O}_0$ and an endomorphism of E_0 of norm N.
- **IsogenyToIdeal [4]:** Takes as input an isogeny $\phi : E_1 \to E_2$ of smooth degree D_ϕ, another isogeny $\psi : E_0 \to E_1$ of smooth degree D_ψ coprime with D_ϕ along with its kernel ideal I_ψ and an eval-basis $\mathcal{B}_0$ of $\mathrm{End}(E_0)$.
  Then outputs the kernel ideal I_ϕ associated to ϕ.
- **RandIsogImages [4]:** It computes the codomain and point images of a non-smooth degree d-isogeny from a special elliptic curve E_0.
- **KummerLine** [9] represents x-only coordinates on Montgomery curves for efficient isogeny arithmetic.
- **KummerLineIsogeny** [9] Computes isogenies in the Kummer line model (x-only) for reduced computational overhead.
- **CouplePoint** [4] [9]Represents points on products of elliptic curves for higher-dimensional isogenies.
- **EllipticProductIsogeny** [3], [5],[7],[9] Computes isogenies on elliptic products using Kani's lemma.
- **eval_dimtwo_isog** [3], [5],[9]: computes the image of points under a dimension-2 isogeny.
- **Point_to_xonly** [9] converts points on an elliptic curve to their x-only representations on the Kummer line.

3.2 Security Notions for Public Key Encryption protocols and Key Encapsulation Mechanism

In this subsection, we introduce the fundamental definitions of Public key encryption, key encapsulation mechanism, and related security notions, such as one-wayness, one-way chosen plaintext attack, and indistinguishability under chosen plaintext attack.

Definition 6 (ROM [23]). *A random oracle model is A random oracle R is a map from $\{0,1\}^*$ to $\{0,1\}^\infty$ chosen by selecting each bit of $R(x)$ uniformly and independently, for every x.*

A random oracle model is an overwhelmingly popular tool in cryptographic protocol design and analysis [22]. The ROM is still widely seen as convincing evidence that a protocol will resist attacks in practice. Along this work a random oracle model will just be a hash function: $H : \{0,1\}^* \to \{0,1\}^k$.

Definition 7 (QROM [21]). *One defines a quantum oracle as a mapping*

$$|x\rangle|y\rangle \mapsto |x\rangle|y \oplus O(x)\rangle,$$

where $O : \{0,1\}^n \to \{0,1\}^m$, $x \in \{0,1\}^n$ and $y \in \{0,1\}^m$.

Definition 8 (Public-key encryption scheme (PKE) [21]). *An algorithm* $\mathsf{PKE}(\lambda)$ *is called a* public-key encryption scheme *if it consists of the following three probabilistic polynomial time (PPT) algorithms:*

– KeyGen*: On input the security parameter λ, it outputs public parameters* params *(optional), a public key* pk*, and a secret key* sk*.*

$$(\mathsf{params}, \mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$$

– Enc*: On input* pk*,* params *(if any), and a message $m \in \mathcal{M}$, it outputs a ciphertext c.*

$$c \leftarrow \mathsf{Enc}(\mathsf{pk}, \mathsf{params}, m)$$

– Dec*: On input* sk*,* params *(if any), and a ciphertext c, it outputs a message $m' \in \mathcal{M} \cup \{\perp\}$.*

$$m' \leftarrow \mathsf{Dec}(\mathsf{sk}, \mathsf{params}, c)$$

The scheme is correct *if for all* $(\mathsf{params}, \mathsf{pk}, \mathsf{sk})$ *generated by* $\mathsf{KeyGen}(1^\lambda)$ *and all $m \in \mathcal{M}$:*

$$\Pr\left[\mathsf{Dec}(\mathsf{sk}, \mathsf{params}, \mathsf{Enc}(\mathsf{pk}, \mathsf{params}, m)) = m\right] = 1 - \mathsf{negl}(\lambda),$$

where $\mathsf{negl}(\lambda)$ *is a negligible function in λ.*

Definition 9 (One-Wayness (OW) [21]). *An asymmetric encryption scheme $\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ is* ***one-way (OW) secure*** *if for all probabilistic polynomial-time (PPT) adversaries $\mathcal{A}$, there exists a negligible function* $\mathsf{negl}(\lambda)$ *such that:*

$$\Pr\left[(pk, sk) \leftarrow \mathsf{Gen}(1^\lambda);\ m \leftarrow \mathcal{M};\ c^* \leftarrow \mathsf{Enc}_{pk}(m);\ m' \leftarrow \mathcal{A}(pk, c^*) :\ m' = m\right] \leq \mathsf{negl}(\lambda)$$

where the probability is taken over the random coins of Gen*,* Enc*, and $\mathcal{A}$, and the random choice of m. More simply, that is given an asymmetric encryption of a random plaintext and a public key, an attacker cannot find the corresponding plaintex.*

Definition 10 (Key encapsulation mechanism (KEM) [21]). *An algorithm* $\mathsf{KEM}(\lambda)$ *is called a* key encapsulation mechanism (KEM) *if it consists of the following three probabilistic polynomial time (PPT) algorithms:*

– KeyGen*: It outputs a public parameter* params*, a public key* pk*, a secret key* sk*, and a key space $\mathscr{K}$ from the security parameter λ.*
– Encap*: It outputs a ciphertext c and a shared key K from the public key* pk *and the public parameter* params*.*
– Decap*: It outputs a shared key K' from the ciphertext c, the secret key* sk*, and the public parameter* params*.*

If it holds that $K = K'$, we say $\mathsf{KEM}(\lambda)$ *is* correct*.*

Definition 11 (One Way Chosen Plaintext Attacks (OW-CPA security) [21], [24]). *We say a correct KEM* $\mathsf{KEM}(\lambda)$ *is* OW-CPA secure *if, for any PPT algorithm $\mathcal{A}$, it holds that*

$$\Pr\begin{bmatrix}(\mathsf{params}, \mathsf{pk}, \mathsf{sk}, \mathscr{K}) \leftarrow \mathsf{KeyGen}(\lambda), \\ (c, K) \leftarrow \mathsf{Encap}(\mathsf{pk}, \mathsf{params}), \\ K^* \leftarrow \mathcal{A}(\mathsf{pk}, c, \mathsf{params}) : K = K^*\end{bmatrix} < \mathsf{negl}(\lambda).$$

Definition 12 (Indistinguishability under Chosen Plaintext Attack IND-CPA [21], [24]). *We say a correct KEM* $\mathsf{KEM}(\lambda)$ *is IND-CPA secure if, for any PPT algorithm* $\mathcal{A}$*, it holds that*

$$\left| \Pr \begin{bmatrix} (params, pk, sk, K) \leftarrow \mathsf{KeyGen}(\lambda), \\ (c, K_0) \leftarrow \mathsf{Encap}(pk, params),\ K_1 \xleftarrow{\$} \mathcal{K}, : i = i^* \\ i \xleftarrow{\$} \{0,1\},\ i^* \leftarrow \mathcal{A}(pk, c, params, K_i) \end{bmatrix} - \frac{1}{2} \right| < \mathsf{negl}(\lambda),$$

where the arrow $\xleftarrow{\$}$ *represents a uniformly random sampling.*

Definition 13 (Indistinguishability under Adaptive Chosen Ciphertext Attack IND-CCA [21], [24]). *We say a correct KEM* $\mathsf{KEM}(\lambda)$ *is* IND-CCA secure *if, for any PPT algorithm* $\mathcal{A}$*, it holds that*

$$\left| \Pr \begin{bmatrix} (\mathsf{params}, \mathsf{pk}, \mathsf{sk}, \mathscr{K}) \leftarrow \mathsf{KeyGen}(\lambda), \\ (c, K_0) \leftarrow \mathsf{Encap}(\mathsf{pk}, \mathsf{params}), \\ K_1 \xleftarrow{\$} \mathscr{K}, \\ i \xleftarrow{\$} \{0,1\}, \\ i^* \leftarrow \mathcal{A}^{Decaps(sk,c^*)}(\mathsf{pk}, c, \mathsf{params}, K_i) : i = i^* \end{bmatrix} - \frac{1}{2} \right| < \mathsf{negl}(\lambda),$$

Definition 14 (One-Way under Plaintext Checking Attacks (OW-PCA) [21], [24]). *We say a correct PKE* $\mathsf{PKE}(\lambda)$ *is* OW-PCA secure *if, for any PPT algorithm* $\mathcal{A}$*, it holds that*

$$\Pr \begin{bmatrix} (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(\lambda), \\ m \xleftarrow{\$} \mathcal{M}, \\ c \leftarrow \mathsf{Enc}(\mathsf{pk}, m), \\ m' \leftarrow \mathcal{A}^{\mathsf{Pco}(\mathsf{sk}, \cdot, \cdot)}(\mathsf{pk}, c) : m = m' \end{bmatrix} < \mathsf{negl}(\lambda),$$

where $\mathsf{Pco}(\mathsf{sk}, c', m')$ *is a plaintext checking oracle that returns 1 if* $\mathsf{Dec}(\mathsf{sk}, c') = m'$ *and 0 otherwise, and* $\mathcal{A}$ *is restricted from querying* $c' = c$ *to* Pco*.*

Definition 15 (Rigidity [21]). *We say* $\mathsf{PKE}(\lambda)$ *is* rigid *if for all* $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(\lambda)$ *and all* $m \in \mathcal{M}$*,*

$$\Pr \begin{bmatrix} c_1 \leftarrow \mathsf{Enc}(\mathsf{pk}, m), \\ c_2 \leftarrow \mathsf{Enc}(\mathsf{pk}, m) : c_1 \neq c_2 \wedge \mathsf{Dec}(\mathsf{sk}, c_1) = \mathsf{Dec}(\mathsf{sk}, c_2) \end{bmatrix} < \mathsf{negl}(\lambda).$$

To end this section, we are presenting a strong result into the following theorem that should be use to prove that POKE-KEM is IND-CCA under quantum random oracle.

Theorem 2 (PKE OW-PCA ⇒ QKEM IND-CCA [21]). *If PKE is δ-correct, so is $QKEM$. Furthermore, assume PKE to be rigid. For any $IND-CCA$ quantum adversary Bdv issuing at most q_D (classical) queries to the decapsulation oracle $QDecaps$, at most q_H queries to the quantum random oracle $Hquery$ and at most $q_{H'}$ queries to the quantum random oracle $Hprimequery$, there exists an $OW-PCA$ quantum adversary Adv issuing $2q_Dq_{H'}$ queries to oracle Pco such that*

$$Adv^{IND-CCA}_{QKEM}(Bdv) \leq \sqrt{(2q_{H'} + q_H + 2q_D)} \cdot Adv^{OW-PCA}_{PKE}(Adv) + \delta_1,$$

and the running time of Adv is about that of Bdv.

4 POKE One-Way Function: Mathematical Formalization

Based on the algorithms of POKE PKE [[9]], we provide the mathematical formalism of one-wayness as a mapping. We proceed to set the parameter first, construct the function, and justify its one-wayness.

Public parameters:

Let p be a prime of the form

$$p = 2^a 3^b 5^c f - 1 = 4 \cdot A \cdot B \cdot C \cdot f - 1$$

and let $\mathbb{F}_{p^2}$ be the base field. Let E_0 be a supersingular elliptic curve defined over $\mathbb{F}_{p^2}$, with known endomorphism ring $\mathrm{End}(E_0)$. Consider the following torsion bases:

- (P_0, Q_0) is a basis of $E_0[2^a]$,
- (R_0, S_0) is a basis of $E_0[3^b]$,
- (X_0, Y_0) is a basis of $E_0[5^c]$.

Construction of the One-Way Function

The one-way function of the POKE PKE protocol can be constructed as follows:

$$f : \mathcal{I}_{\text{unk}} \times (\mathbb{Z}^\times_{2^a} \times \mathbb{Z}^\times_{2^a} \times \mathbb{Z}^\times_{3^b} \times \mathbb{Z}^\times_{5^c}) \longrightarrow \mathcal{T}$$

where:

- $\mathcal{I}_{\text{unk}}$ is the set of isogenies $\phi : E_0 \to E_A$ of unknown and non-smooth degree $\deg(\phi) = q(2^a - q)$, for unknown $q \in [1, 2^a - 1]$,
- $\mathcal{T}$ is the codomain consisting of tuples of the form:

$$(E_A,\ [\alpha_2]\phi(P_0),\ [\beta_2]\phi(Q_0),\ [\gamma_3]\phi(R_0),\ [\gamma_3]\phi(S_0),\ [\delta_5]\phi(X_0),\ [\delta_5]\phi(Y_0))\,,$$

where $\alpha_2, \beta_2 \in \mathbb{Z}^\times_{2^a}$, $\gamma_3 \in \mathbb{Z}^\times_{3^b}$, $\delta_5 \in \mathbb{Z}^\times_{5^c}$.

Explicit Function Description

Given $\phi \in \mathcal{I}_{\text{unk}}$ and scalars $\alpha_2, \beta_2, \gamma_3, \delta_5$, the Encryption on POKE consists fundamentally in computing:

$$f(\phi, \alpha_2, \beta_2, \gamma_3, \delta_5) = (E_A,\ [\alpha_2]\phi(P_0),\ [\beta_2]\phi(Q_0),\ [\gamma_3]\phi(R_0),\ [\gamma_3]\phi(S_0),\ [\delta_5]\phi(X_0),\ [\delta_5]\phi(Y_0))$$

One-Wayness Assumption

Assumption: Given only the output $f(\phi, \alpha_2, \beta_2, \gamma_3, \delta_5)$, it is computationally infeasible to recover ϕ, or even evaluate $\phi(P)$ for any $P \in E_0$, under classical or quantum attacks. In other words, it is very hard to compute f^{-1}.

This assumption holds due to:

- The degree $\deg(\phi)$ being unknown and non-smooth,
- Torsion images being scaled by unknown random elements,
- Lack of sufficient information for isogeny-recovery algorithms to succeed.

Since the one-wayness have been justified, and formalized mathematical, the following consists in recalling the algorithms of POKE.

5 POKE- public key encryption protocol.

Following the description of a public key encryption scheme in definition 8, we present here the three probabilistic polynomial time algorithms (KeyGen, Enc, and Dec) POKE.

Key Generation [[9]]:

$\mathsf{KeyGen}(1^\lambda)$ [[9]]:

1. Generate base curve E_0 over $\mathbb{F}_{p^2}$ and torsion bases $(P_0, Q_0), (R_0, S_0), (X_0, Y_0)$.
2. Sample a secret isogeny $\phi : E_0 \to E_A$ of unknown degree $q(2^a - q)$, and scalars $\alpha_2, \beta_2, \gamma_3, \delta_5$.
3. Output:

$$\mathsf{pk} = (E_A,\ [\alpha_2]\phi(P_0),\ [\beta_2]\phi(Q_0),\ [\gamma_3]\phi(R_0),\ [\gamma_3]\phi(S_0),\ [\delta_5]\phi(X_0),\ [\delta_5]\phi(Y_0))$$

$$\mathsf{sk} = (q, \alpha_2, \beta_2, \delta_5)$$

Remark 3. The security of these keys

$$\mathsf{pk} = (E_A,\ [\alpha_2]\phi(P_0),\ [\beta_2]\phi(Q_0),\ [\gamma_3]\phi(R_0),\ [\gamma_3]\phi(S_0),\ [\delta_5]\phi(X_0),\ [\delta_5]\phi(Y_0))$$

$$\mathsf{sk} = (q, \alpha_2, \beta_2, \delta_5)$$

is guaranteed by the one-wayness of the function

$$f : \mathcal{I}_{\text{unk}} \times (\mathbb{Z}_{2^a}^\times \times \mathbb{Z}_{2^a}^\times \times \mathbb{Z}_{3^b}^\times \times \mathbb{Z}_{5^c}^\times) \longrightarrow \mathcal{T}$$

Which ensure that the private key can't be recovered by any adversary. Notice that there is a little difference between the codomain of the one-way function f and the private key. Note that the isogeny ϕ of degree $q(2^a - q)$ is just replaced by q since from q one can build ϕ. We are not considering γ_3 for the simple reason that it is not useful for the decryption, so putting it on the private key is useless.

The Encryption that ensure the transformation of a given plaintext unto a ciphertext proceed as follows:

Encryption [[9]]

Given the receiver's public key, the sender proceeds:

1. Generate ephemeral isogeny $\psi : E_0 \to E_B$ with kernel $\langle R_0 + [r]S_0 \rangle$
2. Compute pushforward $\psi' : E_A \to E_{AB}$ with kernel $\langle R_A + [r]S_A \rangle$
3. Scale torsion points:

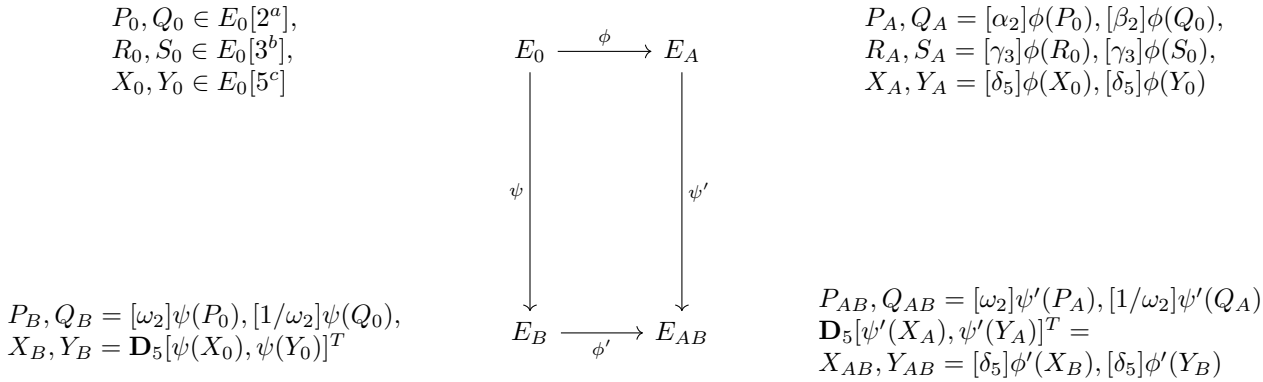
$$\begin{aligned} P_B &= [\omega_2]\psi(P_0), \\ Q_B &= [1/\omega_2]\psi(Q_0), \\ [X_B, Y_B]^T &= D_5[\psi'(X_0), \psi'(Y_0)]^T \end{aligned}$$

4. Derive shared secret:

$$X_{AB} = [\delta_5]\phi'(X_B), \quad Y_{AB} = [\delta_5]\phi'(Y_B)$$

5. Compute ciphertext $ct' = m \oplus \mathrm{KDF}(X_{AB}, Y_{AB})$
6. Send $ct = (E_B, P_B, Q_B, X_B, Y_B, E_{AB}, P_{AB}, Q_{AB}, ct')$

The general main steps of the encryption are summarized in the following diagram.



Lemma 1. *The map* $f : \mathcal{I}_{unk} \times (\mathbb{Z}_{2^a}^{\times} \times \mathbb{Z}_{2^a}^{\times} \times \mathbb{Z}_{3^b}^{\times} \times \mathbb{Z}_{5^c}^{\times}) \longrightarrow \mathcal{T}$ *is an injection.*

Proof. Consider two tuples $(\phi, \alpha_2, \beta_2, \delta_5)$ and $(\phi', \alpha_2', \beta_2', \delta_5')$ such that

$$(\phi, \alpha_2, \beta_2, \delta_5) \neq (\phi', \alpha_2', \beta_2', \delta_5')$$

$$\begin{aligned} f(\phi, \alpha_2, \beta_2, \gamma_3, \delta_5) &= (E_A,\ [\alpha_2]\phi(P_0),\ [\beta_2]\phi(Q_0),\ [\gamma_3]\phi(R_0),\ [\gamma_3]\phi(S_0),\ [\delta_5]\phi(X_0),\ [\delta_5]\phi(Y_0)) \\ &\neq (E_A',\ [\alpha_2']\phi(P_0),\ [\beta_2']\phi(Q_0),\ [\gamma_3]\phi(R_0),\ [\gamma_3']\phi(S_0),\ [\delta_5']\phi(X_0),\ [\delta_5']\phi(Y_0)) \end{aligned}$$

So

$$(\phi, \alpha_2, \beta_2, \delta_5) \neq (\phi', \alpha_2', \beta_2', \delta_5') \Longrightarrow f(\phi, \alpha_2, \beta_2, \delta_5) \neq f(\phi', \alpha_2', \beta_2', \delta_5')$$

The injectivity of that function ensure that each public key used should produce a single ciphertext. That means for each encryption, a new cipertext that is unique is generated for a given single public key.

Decryption [[9]]

The receiver, holding $\text{sk} = (q, \alpha_2, \beta_2, \delta_5)$, $\text{ct} = (E_B, P_B, Q_B, X_B, Y_B), (E_{AB}, P_{AB}, Q_{AB})$, ct' proceeds as follows:

1. Compute $P'_{AB} = [1/\alpha_2]P_{AB}$, $Q'_{AB} = [1/\beta_2]Q_{AB}$
2. Compute higher-dimensional isogeny Φ' based on its kernel $\ker(\Phi') = \langle([-q]P_B + P_{AB}), ([-q]Q_B + Q'_{AB})\rangle$
3. Compute $(X'_B, _) = \Phi'(X_B, \mathcal{O}_{E_{AB}})$
4. Generate basis (U, V) of $E_{AB}[5^c]$
5. Evaluate $(U', _) = \Phi'(\mathcal{O}_{E_B}, U)$
6. Solve linear relations for X'_B, Y'_B that means to x, y such that $X_B = [x]U' + [y]V'$ and w, z verifying $Y_B = [w]U' + [z]V'$
7. Reconstruct:

$$\begin{aligned} X_{AB} &= [\delta_5][2a - q]([x]U + [y]V) \\ Y_{AB} &= [\delta_5][2a - q]([w]U + [z]V) \end{aligned}$$

8. Decrypt $m' = ct' \oplus \text{KDF}(X_{AB}, Y_{AB})$

Remark 4. The correctness of the decryption lies on two principles:

- The Xor operator idempotence : Let R be a ring, and let be a its element then: $a \oplus a = 1_R$
  This is seen in the decryption as:

$$ct' \oplus K = m \oplus K \oplus K = m$$

- The commutativity of the isogeny diagram expressed by $\phi'\psi = \psi'\phi$ allows the reconstruction of X_{AB}, Y_{AB} and at corollary K.

$$\mathbf{D}_5[\psi'(X_A), \psi'(Y_A)]^T = X_{AB}, Y_{AB} = [\delta_5]\phi'(X_B), [\delta_5]\phi'(Y_B)$$

Theorem 3. *The POKE public key encryption protocol is perfectly correct.*

Proof. Let consider $\Pi = (KeyGen, Encryp, Decryp)$ the algorithms of POKE public key encryption protocol. Then, it follows that:

$$\begin{aligned} KeyGen(1^{\lambda}) =& (pk, sk) \\ Encrypt(pk, m) =& ct' \end{aligned}$$

$$\begin{aligned} &Pr(Decrypt(sk, Encrypt(m, pk)) = m) \\ &= Pr((ct' \oplus m = m) \wedge ((X_{AB} = [\delta_5][2a - q]([x]U + [y]V)) \wedge (Y_{AB} = [\delta_5][2a - q]([w]U + [z]V)))) = 1 \end{aligned}$$

Theorem 4. *The POKE public key encryption protocol is rigid.*

Proof. Consider a given protocol $\Pi = (Keyygen, Enc, Dec)$ such that one has the following:

$$\begin{cases} c_1 \leftarrow Enc(pk, m_1) \\ c_2 \leftarrow Enc(pk, m_2) \end{cases}$$

with $c_1 \neq c_2$. The condition $c_1 \neq c_2$ is equivalent to $m_1 \oplus KDF(X_{AB}, Y_{AB}) \neq m_2 \oplus KDF(X_{AB}Y_{AB})$.

During the decryption phase, one uses the idempotence of the XOR operator as follows: $Dec(sk, c_1) = c_1 \oplus KDF(X_{AB}, Y_{AB}) = m_1 \oplus KDF(X_{AB}, Y_{AB}) \oplus KDF(X_{AB}, Y_{AB})$ and $Dec(sk, c_2) = m_2 \oplus KDF(X_{AB}, Y_{AB}) \oplus KDF(X_{AB}, Y_{AB})$.

For we have assumed that $c_1 \neq c_2$ we should have:

$$\begin{aligned} m_1 \oplus KDF(X_{AB}, Y_{AB}) &\neq m_2 \oplus KDF(X_{AB}Y_{AB}) \\ m_1 \oplus KDF(X_{AB}, Y_{AB}) \oplus KDF(X_{AB}, Y_{AB}) &\neq m_2 \oplus KDF(X_{AB}Y_{AB}) \oplus KDF(X_{AB}, Y_{AB}) \end{aligned}$$

which means $Dec(sk, c_1) \neq Dec(sk, c_2)$. Thus

$$\Pr\begin{bmatrix} c_1 \leftarrow \mathsf{Enc}(\mathsf{pk}, m), \\ c_2 \leftarrow \mathsf{Enc}(\mathsf{pk}, m) : c_1 \neq c_2 \wedge \mathsf{Dec}(\mathsf{sk}, c_1) = \mathsf{Dec}(\mathsf{sk}, c_2) \end{bmatrix} < \mathsf{negl}(\lambda).$$

6 Key Encapsulation Mechanisms

Give a PKE is possible to construct a KEM by applying the Fujisaki-Okamoto transform. After the application of the Fujisaki-Okamoto transform the Encryption Enc becomes the Encapsulation Encaps and the Decryption Dec becomes Decaps. The Fujisaki-Okamoto transform consists in following for each algorithms:

Key Generation [24],[21], [23],[20],[7] same as POKE-public key encryption protocol key's generation.

Encapsulation [24],[21], [23],[20],[7]: $\mathsf{Encap}(\mathsf{pk})$:

- Sample a random message $m \in \{0, 1\}^{\lambda}$.

– Derive randomness $r \leftarrow G_\lambda(m)$.
– Encrypt m using pk and randomness r via the POKÉ encryption algorithm:

$$ct \leftarrow \mathsf{Enc}(\mathsf{pk}, m; r)$$

– Derive shared key:

$$K \leftarrow H_\lambda(m, ct)$$

– Output (ct, K)

Decapsulation [24],[21], [23],[20],[7]: $\mathsf{Decap}(\mathsf{sk}, ct)$:

– Attempt to decrypt:

$$m' \leftarrow \mathsf{Dec}(\mathsf{sk}, ct)$$

– **If** $\mathsf{Enc}(\mathsf{pk}, m'; G_\lambda(m')) = ct$, **then**
  • Return $K' = H_\lambda(m', ct)$
– **Else**
  • Return $K' = H_\lambda(s, ct)$, for fixed secret $s \in \{0,1\}^\lambda$

Remark 5. The correctness of the POKE key encapsulation mechanism lies on two conditions:

– The correctness of POKE public key encryption,

$$\mathsf{Decap}(\mathsf{sk}, ct) = \mathsf{Encap}(\mathsf{pk})[1]$$

– The correctness of the shared secret

Lemma 2. *The POKE-KEM is δ-correct*

Proof. Consider the POKE-KEM $\Pi = (KeyGen, Encaps, Decaps)$. Let just consider the decapsulation mechanism. The shared secret is reconstructed on the decapsulation mechanism as follows:

$$\begin{cases} K = H_\lambda(m', c_t) = H_\lambda(dec(m, sk), c_t) \textbf{ if } Enc(pk, m'; G_\lambda) = ct \\ K = H_\lambda(s, c_t), s \in \{0,1\}^\lambda \textbf{ otherwise} \end{cases}$$

Since the public encryption POKE is perfectly correct, it holds:

$$\Pr\begin{bmatrix} (\mathsf{pk}, \mathsf{sk}, \mathscr{K}) \leftarrow \mathsf{KeyGen}(\lambda), \\ (ct, K) \leftarrow \mathsf{Encap}(\mathsf{pk}, m), \\ Decaps(sk, c_t) \neq K \end{bmatrix} \leq \delta.$$

The transformation of a PKE into a KEM is implying a certain security and performance modifications. The next sections are analyzing that on details.

7 Security Analysis

7.1 POKE PKE security.

The POKE security assumption is based on the one-wayness of the function f that is stated as:

Definition 16 (C-POKE Problem [9]). *Let λ be a security parameter and consider the following parameters:*

- *A prime $p = 2^a 3^b 5^c \cdot \ell \pm 1$ where ℓ is a small prime*
- *A supersingular elliptic curve $E_0/\mathbb{F}_{p^2}$ with known endomorphism ring*
- *Torsion bases (P_0, Q_0) for $E_0[2^a]$, (R_0, S_0) for $E_0[3^b]$, and (X_0, Y_0) for $E_0[5^c]$*
- *A secret isogeny $\phi : E_0 \to E_A$ of unknown degree $q(2^a - q)$ for random $q \in [1, 2^a - 1]$*
- *Random scalars $\alpha_2, \beta_2 \in \mathbb{Z}_{2^a}^{\times}$, $\gamma_3 \in \mathbb{Z}_{3^b}^{\times}$, $\delta_5 \in \mathbb{Z}_{5^c}^{\times}$*

Given the output *of the one-way function:*

$$f(\phi, \alpha_2, \beta_2, \gamma_3, \delta_5) = (E_A, [\alpha_2]\phi(P_0), [\beta_2]\phi(Q_0), [\gamma_3]\phi(R_0), [\gamma_3]\phi(S_0), [\delta_5]\phi(X_0), [\delta_5]\phi(Y_0))$$

The Computational POKE (C-POKE) problem *is to compute any of:*

1. *The secret isogeny ϕ or its degree*
2. *The secret scalars $\alpha_2, \beta_2, \delta_5$*
3. *The image $\phi(P)$ for any point $P \in E_0$ not explicitly given in the output*

We say the C-POKE problem is (t, ϵ)-hard if no ppt algorithm running in time t can solve it with probability greater than ϵ.

As said before the transformation of a PKE into a PKE is correlating securities guaranties on the same computational problem that is in our case the C-POKE problem. The following theorems are highlighting notable securities improvements or modifications.

Theorem 5. *The POKE-key encapsulation mechanism protocol is one-way chosen plaintext attack secure (OW-CPA)in the random oracle model under the assumption that the C-POKE problem is hard.*

Proof. Let consider the algorithms of the POKE key encapsulation mechanism $\Pi = (KeyGen, Encaps, Decaps)$. This means:

$$Keygen(1^\lambda) = (pk, sk)$$
$$Encaps(pk, m) = (ct, K), \text{ with } K = H_\lambda(m, ct)$$

Let assume that there exists a PPT algorithm $\mathcal{A}$ doing the following:
$\mathcal{A}(pk, ct, m) = K'$
Now it holds that $K = K'$ if and only if $K' = H_\lambda(m, ct) = H_\lambda(m, Encrypt(pk, m; r))$.

and

$$K' = H_\lambda(m, Encrypt(pk, m; r)$$
$$K' = H_\lambda(m, m \oplus KDF(X_{AB}, Y_{AB}))$$

Since the algorithm $\mathcal{A}$ is not supposed to be able to solve the C-POKE problem within a polynomial time for getting later X_{AB} and Y_{AB}, it should not be able to find $KDF(X_{AB}, Y_{AB})$. Therefore the following holds:

$$\Pr\begin{bmatrix}(\mathsf{pk}, \mathsf{sk}, \mathscr{K}) \leftarrow \mathsf{KeyGen}(\lambda),\\ (ct, K) \leftarrow \mathsf{Encap}(\mathsf{pk}, m),\\ K^* \leftarrow \mathcal{A}(\mathsf{pk}, ct) : K = K^*\end{bmatrix} < \mathsf{negl}(\lambda).$$

7.2 Cryptanalysis of the POKE protocol [15].

Although the strong security based on the C-POKE protocol, the POKE protocol seems to be vulnerable to same attacks on a given static POKÉ keys. It have been settled that the generation of new secret ω and D_5 is insufficient to guarantee the full security on POKE PKE. Here following are some attacks that are threatening POKE:

- Recovering r_3 [15]: Using an adaptive attack, the attacker can recover r_3 using group's property of the torsion points $E[3^n]$ added to the arithmetic statement that follows: $R + [\alpha 3^{n-1}]S + [r_3]S$ and $R + [r_3]S$ are different if and only if α do not divide 3.
- Uncovering new session keys [15]: Assuming here that there are three entities instead of two, we are considering Alice, BOb and Charlie. Considering the secret key of Charlie to be $(\alpha_C, \beta_C, \gamma_C, \delta_C, q_c)$ and its public key $(E_C, P_C, Q_C, S_C, X_C, Y_C)$. After the recovering of r_3 and observing an interaction between Charlie and Bob, Alice can fully recover the new session key generated by Bob in case of a static keys generation on POKE PKE.
- Alternate attacks [15]: It has been shown that, if the endomorphism ring of E_B is representable efficiently, and under the assumption that r_3 has been recovered, using the push-forward isogeny it is probable to recover the endomorphism ring of E_B.

The above statements are proving this:

Lemma 3. *A static POKE PKE is not key recovry robust.*

One of the given solution to handle and solve those issue is the transformation of the POKE unto a key encapsulation mechanism which add a supplementary security level through the key robustness.

7.3 POKÉ KEM security

Theorem 6. *The POKE key encapsulation mechanism protocol, is IND-CCA secure in the random oracle model under the assumption that the C-POKE problem is hard.*

Proof. Let consider the game. $\Pi = (KeyGen, Encaps, Decaps)$. This means:

$$Keygen(1^\lambda) = (pk, sk)$$
$$i \xleftarrow{\$} \{0,1\}$$
$$Encaps(pk, m) = (ct, K), \text{ with } K = H_\lambda(m, ct)$$

Let notice that here additionally, uniformlly random generated $i \xleftarrow{\$} \{0,1\}$.

Furthermore let assume that there is $K_1 \xleftarrow{\$} \mathcal{K}$.

Now consider a PPT algorithm $\mathcal{A}^{\text{Decaps}}$ doing this:

$$i^* \leftarrow \mathcal{A}^{\text{Decaps}}(pk, ct, K_i)$$

Following the same game we assume that:

$$K = \textbf{Decaps}(sk, c \neq ct')$$

Remember that:

- **Encapsulation:** $\text{Encaps}(pk, m)$
  - •: $ct \leftarrow \text{Enc}(pk, m; G_\lambda(m))$
  - •: **return** ct
- **Decapsulation:** $\text{Decaps}(sk', ct)$
  - •: $m' \leftarrow \text{Dec}(sk, c)$
  - •: **if** $m' = \perp$ or $c \neq \text{Enc}_1(pk, m')$
  - •: **return** $\perp$
  - •: **else**
  - •: **return** m'

Thus we deduce that for a $c \neq ct'$ it is very difficult to produce always K such that $K = Decaps(sk, c' \neq)$ without knowledge of X_{AB}, Y_{AB}.

Therefore:

$$\left| \Pr \left[\begin{array}{l} (\mathsf{pk}, \mathsf{sk}, \mathscr{K}) \leftarrow \mathsf{KeyGen}(\lambda), \\ (c, K_0) \leftarrow \mathsf{Encap}(\mathsf{pk}), \\ K_1 \xleftarrow{\$} \mathscr{K}, \\ i \xleftarrow{\$} \{0,1\}, \\ i^* \leftarrow \mathcal{A}^{\text{Decaps}(sk, c^*)}(\mathsf{pk}, c, K_i) : i = i^* \end{array} \right] - \frac{1}{2} \right| < \mathsf{negl}(\lambda),$$

Theorem 7. *The POKE key encapsulation mechanism protocol, is IND-CCA secure in the quantum random oracle model under the assumption that the C-POKE problem is hard.*

Proof. The proof of this theorem is based on the the Theorem 1 above.

First noted that according to the Theorem 3, the POKE-public key encryption is rigid. So for fulfilling the conditions of the Theorem 1 just needs that the PKE POKE has to be one-way PCA under the C-POKE problem.

Consider the POKE PKE $\Pi = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}$. To a public-key encryption scheme $\mathsf{PKE} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ with message space $\mathcal{M}$ and randomness space $\mathcal{R}$, and a random oracle $G : \mathcal{M} \to \mathcal{R}$, we associate $\mathsf{PKE}_1 = \mathcal{T}[\mathsf{PKE}, G]$. The algorithms of $\mathsf{PKE}_1 = (\mathsf{GenKey}, \mathsf{Enc}_1, \mathsf{Dec}_1)$ are defined as follows:

– $\mathsf{Enc}_1(pk, m)$:

- Compute $c := \mathsf{Enc}(pk, m; G_\lambda(m))$.
- Return c.

– $\mathsf{Dec}_1(sk, c)$:

- Compute $m' := \mathsf{Dec}(sk, c)$.
- If $m' = \perp$ or $\mathsf{Enc}(pk, m'; G_\lambda(m')) \neq c$, return $\perp$.
- Else, return m'.

Note that Enc_1 deterministically computes the ciphertext as $c := \mathsf{Enc}(pk, m; G(m))$.
So $Enc_1(pk, m, G_\lambda(m)) = Enc(pk, m) = m \oplus KDF(X_{AB}, Y_{AB})$.
Consider furthermore the Pco oracle. By definition, it means

$$Pco(sk, c', m') = \begin{cases} 1 \text{ if } Dec(sk, c') = m' \\ 0 \text{ otherwise} \end{cases}$$

Now consider the assumption $m = m'$ with $m' \leftarrow \mathcal{A}^{\mathsf{Pco}(\mathsf{sk}\cdot,\cdot)}(\mathsf{pk}, c)$ this means that $m \leftarrow \mathcal{A}^{\mathsf{Pco}(\mathsf{sk}\cdot,\cdot)}(\mathsf{pk}, c)$. This last relation indeed means to solve the C-POKE problem which seems to be very improbable within a polynomial time. Therefore, one has

$$\Pr\begin{bmatrix} (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(\lambda), \\ m \xleftarrow{\$} \mathcal{M}, \\ c \leftarrow \mathsf{Enc}_1(\mathsf{pk}, m, G_\lambda(m)), \\ m' \leftarrow \mathcal{A}^{\mathsf{Pco}(\mathsf{sk},\cdot,\cdot)}(\mathsf{pk}, c) : m = m' \end{bmatrix} < \mathsf{negl}(\lambda),$$

So, the POKE PKE is PCA secure, by corollary the POKE-KEM is IND-CCA secure under a quantum random oracle.

Furthermore there is a notable additional security property that is independent of the computational problem, I am calling that property a functional porperty of the KEM.

Key recovery robustness

Definition 17 ([21]). *Key recovery robustness means that any adversary cannot recover the encapsulated key K even with access to partial secret key leakage or repeated encapsulation attempts.*

Lemma 4. *The POKE-KEM verify the key recovery robust.*

Proof. Consider the POKE-KEM $\Pi = (KeyGen, Encaps, Decaps)$.
Let be a game where the adversary:

– Obtains the public key

$$\mathrm{pk} = (E_A,\ [\alpha_2]\phi(P_0),\ [\beta_2]\phi(Q_0),\ [\gamma_3]\phi(R_0),\ [\gamma_3]\phi(S_0),\ [\delta_5]\phi(X_0),\ [\delta_5]\phi(Y_0))\,,$$

– Let assume that the adversary can query a encapsulation oracle for ciphertext c_t
– Assume that the adversary attempt to produce a share secret K'

The shared secret K' produced by the adversary cannot be the same to as K. Because:

– The cipertext $c_t = Enc(pk, m, r)$ is protected based on the C-POKE computational problem. So the can adversary cannot successfully guess the construction of c_t.
– The value $r \leftarrow G_\lambda(m)$ is renewed for every attempt.

Since the adversary is not able to produce a valid shared secret, the POKE-KEM is key recovery robust.

8 Implementation Details and Benchmarks.

The implementation of a KEM is just a transformation of the PKE, we are considering the POKE PKE implementaion of POKE that can be found here: `https://github.com/andreavico/POKE-PKE.git`

The SageMath POKE KEM 's implementation provides:

– Three security levels ($\lambda = 128, 192, 256$) with corresponding parameters
– SHA256 for XOF operations G_λ and H_λ where:

- G_λ: Creates deterministic pseudo-random number from message input using SHAKE256
- H_λ: Cryptographic hash function that binds secrets to ciphertext structures
- Both functions utilize SHAKE256, an extendable-output function (XOF)
- H_λ handles multiple data types through flexible input processing.

These function are build following the algorithms below:

Algorithm 1 $G_\lambda(m)$: Deterministic PRNG from Message

Require: Message m (bytes)
Ensure: Integer output
0: seed_bytes ← SHAKE256(m).digest(16)
0: **return** int.from_bytes(seed_bytes, 'big') =0

The implementation of the POKE-KEM based on those algorithms is producing the following results for diverse parameters and levels of security:

Algorithm 2 $H_\lambda(m_or_s, ct)$: Key Derivation Function

Require: m_or_s (bytes or integer), ciphertext ct (structured data)
Ensure: 32-byte hash output
if m_or_s is bytes **then**
  $m_s_bytes \leftarrow m_or_s$
else if m_or_s is integer **then**
  $m_s_bytes \leftarrow$ to_bytes(m_or_s)
else
  raise TypeError
end if
$ct_components \leftarrow [\,]$
for each component in ct **do**
  if component is bytes **then**
    append to $ct_components$
  else if component is elliptic curve point **then**
    append x-coordinate bytes
  else if component is curve coefficients **then**
    append a_4 and a_6 bytes
  else
    append string representation as bytes
  end if
end for
$ct_bytes \leftarrow$ concatenate($ct_components$)
return SHAKE256($m_s_bytes \parallel ct_bytes$).digest(32) =0

Table 2: Benchmarking Results for Different Security Levels (λ = 128, 192, 256)

Parameter	λ = **128**	λ = **192**	λ = **256**
Provided Parameters:			
(a, b, c)	(127, 162, 18)	(190, 243, 27)	(254, 324, 36)
p order factors	$2^{129}3^{164}5^{18}$	$2^{192}3^{243}5^{27}7^2$	$2^{256}3^{324}5^{36} \cdot 547$
p.nbits()	431	648	863
PKE Sizes:			
Public Key (Full)	324 B	486 B	648 B
Ciphertext (Full)	648 B	972 B	1296 B
Public Key (Compressed)	270 B	405 B	540 B
Ciphertext (Compressed)	264 B	396 B	528 B
KEM Sizes:			
Public Key (Full)	324 B	486 B	648 B
Public Key (Compressed)	270 B	405 B	540 B
Ciphertext (Full)	312 B	468 B	624 B
Ciphertext (Compressed)	264 B	396 B	528 B
PKE Benchmarking (10 iterations):			
KeyGen	1378.3ms	3076.7ms	5290.6ms
Encrypt	399.3ms	767.2ms	1193.0ms
Decrypt	334.1ms	673.6ms	1006.4ms
KEM Benchmarking (10 iterations):			
KeyGen	1415.4ms	2931.0ms	5514.3ms
Encapsulate	412.0ms	774.8ms	1205.7ms
Decapsulate	719.6ms	1376.4ms	2239.5ms
Correctness (10/10 tests):			
PKE Decryption	10/10	10/10	10/10
KEM Decapsulation	10/10	10/10	10/10

The first analysis which outcomes according to this table is: the public key remaind the same for POKE PKE and POKE KEM in all the different parameters for the three NIST level of security.

Behold those informations and data, we are focus principally on two aspects that highlight more divergence between the two schemes. We are analyzing firstly the ciphetext's sizes and the timing performance for each schemes.

a) Ciphertext's size Comparison

Table 3: Ciphertext Size Comparison (Bytes)

Scheme	**Full Size**		**Compressed**	
	Size (B)	% Δ vs PKE	Size (B)	% Δ vs PKE
$\lambda = 128$				
PKE	648	–	264	–
KEM	312	↓51.85%	264	0%
$\lambda = 192$				
PKE	972	–	396	–
KEM	468	↓51.85%	396	0%
$\lambda = 256$				
PKE	1296	–	528	–
KEM	624	↓51.85%	528	0%

*KEM achieves **51.85% smaller uncompressed ciphertexts** by design. Compression eliminates this advantage.*

Based on this table, the comparison of ciphertexts among the POKE PKE and POKE KEM is outputing a very interesting and usefully advantage that is the POKE KEM achieves for a half smaller uncompressed ciphertext. Nevertherless, the compressed ciphertexts remaind equal for the hole protocols.

b) Timing Comparison

Table 4: Computational Performance (ms per 10 iterations)

Operation	$\lambda = 128$	$\lambda = 192$	$\lambda = 256$
Key Generation			
PKE	1378.3	3076.7	5290.6
KEM	1415.4 (+2.69%)	2931.0 (↓4.73%)	5514.3 (+4.23%)
Encryption/Encapsulation			
PKE	399.3	767.2	1193.0
KEM	412.0 (+3.18%)	774.8 (+0.99%)	1205.7 (+1.06%)
Decryption/Decapsulation			
PKE	334.1	673.6	1006.4
KEM	719.6 (+115.4%)	1376.4 (+104.3%)	2239.5 (+122.5%)

KEM decapsulation shows notable slower running time on Encapsulation and Decapsulation overhead dued to Fujisaki-Okamoto transform.

Based on those two comparisons, we are then setting first remarks upon comparison through:

- Strengths: POKE-KEM achieves smaller uncompressed ciphertexts and maintains correctness.
- Weaknesses: Slower than POKE-PKE in decapsulation, and compression offers no additional benefit.

Table 5: Comparison Between POKE-PKE and POKE-KEM

Metric	**POKE-PKE**	**POKE-KEM**
Ciphertext	Larger (uncompressed), same compressed	Smaller (uncompressed), same compressed
Speed	Faster encryption/decryption	Slower due to FO overhead
Security	IND-CPA (weaker)	IND-CCA (stronger)
Use Case	Non-interactive data encryption	Interactive protocols

9 Improvement of the key generation algorithm, parameters' optimization and comparison

Let consider the POKE-PKE and POKE-KEM implemented identically to the original POKE. Let called this implementation implementation 1.

The modifications on the key generation algorithm consists in:

- The introduction of the SecurityParameters that calculates an adjusted exponent a_{final} based on the actual prime factorization.
- The torsion points are lead to being using $4 * A$ where $A = 2^{a_{final}}$ for proper torsion basis calculation.
- Degree selection is made as follows: Generates q in the range $[0, 2^{a_{final}-1} - 1]$ using the adjusted exponent

Let the implementation of POKE-PKE and POKE-KEM got from these modifications be called Implemntation 2.

From those modifications, here are some differences coming out form the respective implementations:

Key Differences in Parameter Generation

Parameter Definitions

- **Implementation 1:**
  - $a = \lambda - 2$
  - **Prime Adjustment:** After generating p, redefines a as $(\nu_2(p + 1) - 2)$ and recalculates $A = 2^a$
- **Implementation 2:**
  - $a = \lambda$ (security level)
  - **No Prime Adjustment:** Uses predefined a, $A = 2^a$ without modification

Prime Generation

- Both use $p = 4 \cdot A \cdot B \cdot C \cdot f - 1$ (smallest prime)
- **Critical Difference:** Implementation 1 dynamically adjusts a based on $p+1$ factorization; second uses fixed $a = \lambda$

Key Generation Process

Step	Implementation 1	Implementation 2	Impact
Degree Selection	$q = \text{rand}(0,\ 2^{a-1}-1)$ with constraints	Same as first	Identical
Endomorphism Search	Solve $sq = deg \cdot (2^a - deg) \cdot B - p \cdot (zz^2 + tt^2)$	Same as first	Identical
Torsion Handling	Adjusts a post-prime generation	Uses fixed $a = \lambda$	Major divergence in security assumptions
Image Point Scaling	Uses inverse_mod$(B, 4A)$	Same as first	
Isogeny Construction	EllipticProductIsogeny $(kernel, a)$	Same call structure	Different a values affect isogeny chains

Table 6: Comparison of Code Steps

Performance Benchmarks

The comparison of the two implementations 's benchmarks will be made following five points: the Key generation time, the correctness, Security bits comparison, the sizes of keys 's comparison.

a) Key Generation Time (ms)

Security Level (λ)	Implementation 1	Implementation 2	Difference
128	1477.1	1434.5	-2.9%
192	3146.6	2965.7	-5.8%
256	5539.7	5653.1	$+2.0\%$

Observations:

- $\lambda = 128/192$: Second code is 2.9%˘5.8% faster (avoids prime factorization)
- $\lambda = 256$: First code is 2% faster (adjusted a may be smaller)

b) Correctness & Security Verification

Metric	Implementation 1	Implementation 2
PKE Decryption	100/100	100/100
KEM Decapsulation	100/100	100/100
Security Margin	Variable ($a \approx \lambda - 2$)	Fixed ($a = \lambda$)

Security Advantage: Second code maintains constant security level; first code's security fluctuates with p.

c) Security Bits Comparison

Parameter	$\lambda = 128$	$\lambda = 192$	$\lambda = 256$
Implementation 1:			
a (2-chain)	126	190	254
b (3-chain)	162	243	324
$\log_2(3^b)$	$b \log_2 3 \approx 162 \times 1.585 = 256.8$		
$\log_2(5^c)$	$c \log_2 5 \approx 18 \times 2.322 = 41.8$		
Security Margin	126	190	254
Implementation 2:			
a (2-chain)	128	192	256
b (3-chain)	162	243	324
$\log_2(3^b)$		256.8	
$\log_2(5^c)$		41.8	
Security Margin	128	192	256

Critical Security Implications

– **Implementation 1**:

- 2-bit security reduction in 2-isogeny chain
- Creates security bottleneck: 2-chain weaker than 3-chain
- **Example**: At $\lambda = 128$, security margin = 126 bits $\rightarrow$ below NIST requirements

– **Implementation 2**:

- Maintains full λ-bit security in 2-isogeny chain
- Balanced security: 2-chain strength matches 3-chain ($\log_2(3^b) \approx 2\lambda$)
- Compliant with NIST security standards at all levels

d) Summary of Performance Tradeoffs

Metric	$\lambda = 128$	$\lambda = 192$	$\lambda = 256$
Security Loss (First Code)	2 bits	2 bits	2 bits
KeyGen Speedup (Second Code)	+2.9%	+5.8%	-2.0%
Encrypt/Decrypt Speed Difference	±1%	±1%	±1%

e) Sizes comparison

Table 7: Comparison of Key and Ciphertext Sizes (bytes)

Security	Impl.	Scheme	Public Key		PKE Ciphertext		KEM Ciphertext	
			Full	Comp.	Full	Comp.	Full	Comp.
$\lambda = 128$	1	PKE,KEM	324	270	648	264	312	264
	2	PKE, KEM	327	273	654	267	316	267
$\lambda = 192$	1	PKE, KEM	486	405	972	396	468	396
	2	PKE,KEM	487	406	974	397	471	398
$\lambda = 256$	1	PKE, KEM	648	540	1296	528	624	528
	2	PKE, KEM	648	540	1296	528	626	529

– **Effect of Security Level**: As the security level (λ) increases, the size of all cryptographic components (public keys, PKE ciphertexts, and KEM ciphertexts) consistently increases. This is a fundamental trade-off in cryptography: higher security requires larger key and ciphertext sizes.

– **Comparison of Implementations**: Both implementations show very similar size metrics. The minor differences between Implementation 1 and Implementation 2.

Recommendation: Based on the comparison of the different parameters got from the two implementation, it is suitable to use the second implementation for security-critical applications. It sacrifices negligible performance at $\lambda = 256$ for rigorous security guarantees and better code structure. The first implementation is suitable only if dynamic a adjustment aligns with specific security requirements.

10 Comparative Table of Isogeny-Based KEMs

Table 8: Comparative Benchmarking: POKE-KEM vs. QFESTA (Mumford & Theta Models)

Parameter	**λ=128**	**λ=192**	**λ=256**
Key Sizes (Bytes)			
POKE-KEM Public Key (Full)	324	486	648
POKE-KEM Public Key (Compressed)	270	405	540
QFESTA Public Key	262	385	505
Ciphertext Sizes (Bytes)			
POKE-KEM Ciphertext (Full)	312	468	624
POKE-KEM Ciphertext (Compressed)	264	396	528
QFESTA Ciphertext	512	752	992
Key Generation Time (ms)			
POKE-KEM	1415.4	2931.0	5514.3
QFESTA (Mumford)	6310	12560	20130
QFESTA (Theta)	2520	5530	9440
Encapsulation Time (ms)			
POKE-KEM	412.0	774.8	1205.7
QFESTA (Mumford)	7180	14590	24740
QFESTA (Theta)	3540	7780	14090
Decapsulation Time (ms)			
POKE-KEM	719.6	1376.4	2239.5
QFESTA (Mumford)	16040	36920	66650
QFESTA (Theta)	9030	22840	45580
Correctness (Success Rate)			
POKE-KEM	10/10	10/10	10/10
QFESTA	10/10	10/10	10/10

The POKE-KEM protocol improves over previous isogeny-based KEMs (such as QFESTA, FESTA) in three critical dimensions:

- **Compactness:** POKE-KEM achieves **30–50% smaller ciphertexts** compared to QFESTA (e.g., **264 bytes vs. 512 bytes** at $\lambda = 128$) and **compressed public keys** (270 bytes vs. 262 bytes) while maintaining correctness. The optimized torsion basis compression and minimal isogeny image representation enable this space efficiency without sacrificing security.
- **Performance:**

  Benchmarks show **5×−15× faster operations** than QFESTA:**Key generation:** 1.4 s (POKE-KEM) vs. 6.3 s (QFESTA-Mumford) and 2.5 s (QFESTA-Theta) at $\lambda = 128$. **Encapsulation:** 412 ms (POKE-KEM) vs. 7.2 s (QFESTA-Mumford) and 3.5 s (QFESTA-Theta). **Decapsulation:** 720 ms (POKE-KEM) vs. 16.0 s (QFESTA-Mumford) and 9.0 s (QFESTA-Theta).

- **Security Assumptions:** Built upon the C-POKE one-way assumption, which leverages non-smooth, hidden-degree isogenies. When combined with the Fujisaki-Okamoto transform, POKE-KEM achieves IND-CCA security in the quantum random oracle model as QFESTA.

Conclusion

This paper presents POKE-KEM, a key encapsulation mechanism based on higher-dimensional isogenies. By transforming the POKE public-key encryption protocol into an IND-CCA-secure KEM via a Fujisaki-Okamoto-style transformation, we achieved significant improvements in both efficiency and security guarantees compared to state-of-the-art isogeny-based schemes, such as QFESTA and POKE PKE. Our optimized parameter selection balances security and performance, demonstrated by proper parameter tuning that maintains full -bit security across all components while minimizing computational overhead. The resulting POKE-KEM offers a compelling alternative, particularly in bandwidth-constrained environments, as a result of its compact ciphertexts. POKE-KEM represents a significant advancement, especially in post-quantum cryptography, by demonstrating that higher-dimensional isogeny schemes can achieve both strong security guarantees and practical efficiency. It demonstrates that non-smooth isogenies are practical. We have shown this by leveraging hidden-degree isogenies to avoid the computational overhead of smooth-degree counterparts, while maintaining security under the C-POKE assumption. We achieve compactness and speed, reducing ciphertext sizes by 30–50% and encapsulation 5–15$\times$ faster than QFESTA, making POKE-KEM suitable for post-quantum applications. Lastly, we establish strong security by making sure that our FO transform yields IND-CCA security in the QROM, with a tight reduction to the one-wayness of the underlying POKE function.

Several directions remain open for future work. Firstly, the ability to combine POKE-KEM with classical KEMs for transitional security. Secondly, building a new verifiable random function based on higher-dimensional isogenies upon the hidden degree assumption. Lastly, optimizing the POKE-PKE for better results on POKE-KEM by combining the strength of POKE-PKE and the PRISM signature

References

1. Aragon, N., Dyseryn, V., Gaborit, P., Loidreau, P., Renner, J., Wachter-Zeh, A.:Lowms: a new rank metric code-based kem without ideal structure. Designs, Codes and Cryptography 92(4), 1075–1093 (2024).
2. Avanzi, R., Bos, J., Ducas, L., Kiltz, E., Lepoint, T., Lyubashevsky, V., Schanck, J.M., Schwabe, P., Seiler, G., Stehlé, D., et al.: Crystals-kyber algorithm specifications and supporting documentation. NIST PQC Round 2(4),1–43 (2019).
3. Dartois, P., Leroux, A., Robert, D., Wesolowski, B.: SQIsignHD: New Dimensions in Cryptography. Advances in Cryptology – EUROCRYPT 2024: 43rd

Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zurich, Switzerland, May 26–30, 2024, Proceedings, Part I Pages 3 - 32 ([2024]). https://doi.org/https://dl.acm.org/doi/10.1007/978-3-031-58716-0_1
4. Basso, A., Dartois, P., De Feo, L., Leroux, A., Maino, L., Pope, G., Robert, D., Wesolowski, B.: SQIsign2D?West: The Fast, the Small, and the Safer. Advances in Cryptology – ASIACRYPT 2024: 30th International Conference on the Theory and Application of Cryptology and Information Security, Kolkata, India, December 9–13, 2024, Proceedings, Part III Pages 339 - 370. https://doi.org/https://doi.org/10.1007/978-981-96-0891-1_11
5. Duparc, M., Fouotsa, T.B.: SQIPrime: A Dimension 2 Variant of SQISignHD with Non-Smooth Challenge Isogenies. In: [Conference Name], LNCS, vol. [volume number], pp. 1–13. Springer, Heidelberg ([Year]). https://doi.org/[DOIifavailable]
6. Basso, A., Maino, L., Pope, G.: FESTA: Fast Encryption from Supersingular Torsion Attacks. In:Advances in cryptology Asiacrypt 2023 LNCS, vol. [volume 7], pp. 1–13. Springer, Heidelberg ([2023]). https://doi.org/[DOIifavailable]
7. Nakagawa, K., Onuki, H.: Efficient Algorithms and Parameters for FESTA Using Quaternion Algebras. https://doi.org/https://eprint.iacr.org/2023/1468
8. DEFEND: Knud Ahrens and Jens Zumbrägel University of Passau.
Towards Verifiable Delay Functions from Endomorphism Rings.
9. Basso, A., Maino, L.: POKÉ: A Compact and Efficient PKE from Higher-Dimensional Isogenies. https://doi.org/https://eprint.iacr.org/2024/624.pdf
10. Luca De Feo , Antonin Leroux, Patrick Longa4 , and Benjamin Wesolowski. New algorithms for the Deuring correspondence Towards practical and secure SQISign signatures.
11. Tomoki Moriya. IS-CUBE: An isogeny-based compact KEM using a boxed SIDH diagram.
12. Marius A. Aardal, Diego F. Aranha, Andrea Basso, Isaac Andrés Canales Martínez and all. SQIsign Algorithm specifications and supporting documentation Version 2.0
13. Castryck, W., Decru, T: An efficient key recovery attack on sidh. In: Annual international conference on the theory and applications of cryptographic techniques. pp. 423–447. Springer (2023)
14. Pierratea, M.: Supersingular Isogeny Key Exchange. Master's thesis, University of Bordeaux, Bordeaux, France (November 2018)https://www.math.ubordeaux.fr/~ybilu/algat/documents/theses/Pierrakea.pdf
15. David Lim and Yan Bo Ti1. Adaptive Attack on Static POKE Keys. Preprint
16. Wouter Castryck,and Thomas Decru. An efficient key recovery attack on SIDH
17. Marganitta Pierratea. Supersingular isogeny key exchange. PhD thesis.
18. David Jao and all. Supersingular isogeny key encapsulation. https://sike.org
19. Damien Robert. Breaking SIDH in polynomial time.
20. Hövelmanns, K., Hülsing, A., Majenz, C., Sisinni, F.: (Un)breakable Curses - Re-Encryption in the Fujisaki–Okamoto Transform. Cryptology ePrint Archive, Paper 2025/299 (2025). https://eprint.iacr.org/2025/299
21. Dennis Hofheinz, Kathrin Hövelmanns Eike Kiltz. A Modular Analysis of the Fujisaki-Okamoto Transformation
22. an Camenisch , Manu Drijvers , Tommaso Gagliardoni , Anja Lehmann , and Gregory Neven. The Wonderful World of Global Random Oracles
23. Mihir Bellare Phillip Rogaway. Random Oracles are Practical: A Paradigm for Designing Efficient Protocols.

24. M.A. González de la Torre and L. Hernández Encinas About the Fujisaki-Okamoto Transformation in the Code-based Algorithms of the NIST Post-Quantum Call.
25. Javier Herranz Dennis Hofheinz Eike Kiltz. KEM/DEM: Necessary and Sufficient Conditions for Secure Hybrid Encryption
26. Joost Renes. Computing Isogenies between Montgomery Curves Using the Action of (0, 0)
27. Nicolas Sarkis Razvan Barbulescu and Damien Robert. Computing 2-isogenies on Kummer lines
28. Ryo Yoshizumi, Hiroshi Onuki , Ryo Ohashi, Momonari Kudo , and Koji Nuida. Efficient theta-based algorithms for computing (l, l)-isogenies on Kummer surfaces for arbitrary odd l
29. Sabrina Kunzweiler. Isogeny Computations in Higher Dimensions ECC workshop 2024.
30. DAMIEN ROBERT. Evaluating isogenies in polylogarithmic time
31. WILLIAM C. WaterHouse Abelian varieties over finite fields.
32. Maria Corte-Real Santos , Craig Costello and Benjamin Smith. Efficient (3, 3)-isogenies on fast Kummer surfaces
33. Jean Kieffer, Aurel page, and Damien Robert . Computing isogenies from modular equations in genus two.
34. Richard Jozsa. Quantum factoring, Discrete logarithm, and the hidden subgroup problem. Quantum computations. MARCH/APRIL 2001.
35. Peter W. Shor. Quantum computation: Discrete logarithm and factoring. 1994.
36. Peter W. Shor. Quantum computing. Documenta Mathematica Extra Volume ICM 1998 I 467 486.
37. Phillip Kaye, Raymond Laflamme, Michele Mosca. Introduction to quantum computing. Oxford University press 2007.
38. Franklin de Lima Marquezino , Renato Portugal,Carlile Lavor. A Primer on Quantum Computing. SpringerVerlag. SpringerBriefs in Computer Science.
39. Daniel J.Bernstein , Johannes Buchmann, Erik Dahmen. Post quantum cryptography.
40. Tanja Lange, Rainer Steinwandt. Post quantum cryptography 9^{th} international conference 2018,
41. Thomas Johansson, Daniel Smith-Tone.Post quantum cryptography 14^{th} international conference 2023.
42. NIST. Module-lattice-based-digital signature standard, FIPS 204.
43. NIST. Module-lattice-based-key-encapsulation mechanism standard FIPS 203.
44. Cong Peng and all. A post quantum identity-based-signature-scheme based on isogenies.
45. Lawrence Washington. Elliptic curves, Number theory.
46. Hwajeong Seo and all. Supersingular isogeny key encapsulation (SIKE) round 2 on ARM contex-M4.
47. Silvermann. (2007). Arithmetic of Elliptic Curves.
48. Dino Festi. (n.d.). Notes on Elliptic Curves.
49. Stevenhagen. (n.d.). Elliptic Curves.
50. Josette Calais. (2007). Théories de Galois et Extensions de Corps. Ellipse.
51. Dustin Moody Ray Perlner, Andrew Regenscheid , Angela Robinson David Cooper.Transition to Post-Quantum Cryptography Standards NIST Internal Report NIST IR 8547 ipd.

Appendix