Updatable Public-Key Encryption from FESTA

Andrea Basso[1], Tako Boris Fouotsa[2], Fatna Kouider[3], Péter Kutas[3,4], Luciano Maino[4], and Laurane Marco[5]

[1] IBM Research Europe, Zürich, Switzerland,
andrea.basso@ibm.com
[2] The University of Manchester, Manchester, UK
research@borisfouotsa.com
[3] Eötvös Loránd University, Budapest, Hungary
fatnakouider@inf.elte.hu
[4] University of Birmingham, Birmingham, UK
kutasp@gmail.com, l.maino@bham.ac.uk
[5] EPFL, Lausanne, Switzerland
marco.laurane@outlook.com

Abstract. Updatable public-key encryption (UPKE) is a cryptographic primitive that was proposed for secure messaging to provide forward secrecy in public-key settings. It extends standard public-key encryption with a key-update mechanism that lets anyone update a receiver's public key and issue a corresponding token for updating the secret key. Unlike traditional forward secrecy where all past messages should remain secure after a key leakage, UPKEs guarantee security only as long as at least one honest update has occurred. While classically-secure efficient instantiations of UPKE are known from Diffie-Hellman assumptions, constructing *efficient post-quantum secure* UPKE schemes with *unbounded* updates remains an active research area. For example, prior lattice-based constructions have update tokens of size 1486 KiB. While isogeny group action based constructions offer acceptable sizes, their quantum security is still subject to debate. Designing efficient isogeny-based UPKEs, which are not based on group actions, is still an open problem.
In this work, we propose a new isogeny-based UPKE that relies on a dimension-four version of the FESTA public-key encryption scheme. It allows an unbounded number of updates. Moreover, we provide a formal security proof based on the CIST^2 problem introduced in FESTA [8, Problem 8] and on the IND-KLPT assumption which states that outputs of the KLPT algorithm are indistinguishable from random isogenies of the same degree. Our UPKE is compact, with public keys, ciphertexts and update tokens of size 0.4 KiB, 2.13 KiB and 2.13 KiB respectively; hence being the second most compact post-quantum UPKE after isogeny group action based ones. When compared to SILBE, the prior isogeny-based UPKE which did not rely on isogeny group actions, our UPKE is several orders of magnitude more efficient.

Keywords: Isogenies, Updatable Public-Key Encryption, FESTA

This is an extended version of the paper accepted at Asiacrypt 2026. The only difference is that this version includes Appendix A recalling basic definitions for PKEs and UPKEs.

1 Introduction

Updatable Public-Key Encryption (UPKE) is an extension of public-key encryption that was proposed in the setting of secure messaging. A necessary property of secure messaging is *forward secrecy*, which ensures that in a scenario where a secret key is exposed at a certain point of time, all messages exchanged before that key leakage remain secure.

This notion is easy to achieve from a primitive called a *ratchet*, which can be instantiated from both symmetric and Diffie-Hellman assumptions, but it requires interaction at every update. An equivalent notion would therefore be desirable from public-key encryption, allowing any party to update a public key and removing the need for interaction. Forward-secure public-key encryption as a notion was proposed and realized in the form of a mechanism that created key pairs which could and should be synchronously updated. However, no efficient constructions have been proposed yet. Also, requiring synchronicity does not seem to be a realistic assumption in a messaging context.

Updatable public-key encryption (UPKE) was therefore put forward as an alternative by Jost et al. [36] and refined by Alwen et al. [3]. The idea is to introduce an asynchronous key update mechanism, where the sender can update the receiver's public key and send them an encrypted update token that will allow them to update their corresponding secret key. The security guarantee is slightly more relaxed than forward security in the sense that exposing a secret key preserves the security of previous messages, as long as an honest update was performed at some point before the exposure.

Efficient constructions of UPKEs were proposed by the original authors [36,3] from discrete logarithm assumptions. However, achieving post-quantum security is now a necessary goal in messaging applications, not only in theory but also in practice, as shown for example with Signal's introduction of a post-quantum ratchet.[6] Building efficient quantum-resistant UPKEs is therefore a necessary step towards efficient and secure messaging protocols.

It is important to note that UPKEs differ significantly from Updatable Encryption, (UE) which is a primitive proposed in [12]. UE is a symmetric-key primitive where a server can update ciphertexts without the knowledge of the secret key (and transmit the update token to the clients who can then update their secret keys locally). The main application of UE is secure storage where the updatability is helpful in forward and post-compromise security. One major difference between these protocols is that the main point of UE is that ciphertexts can be updated, while, in general, there are no requirements in a UPKE about the updatability of ciphertexts. We refer the reader to the recent survey paper by Manulis, Slamanig and Valbusa [41, Section 1] for more information about the classification of UEs and UPKEs.

Related work. We now proceed to survey relevant work in the area of post-quantum updatable public key encryption.

[6] https://signal.org/blog/spqr/

The first post-quantum updatable public-key encryption schemes were proposed from lattice-based assumptions, namely by Dodis et al. and Haidar et al. [27,34]. However, both of these schemes only support a bounded number of updates, due to the inherent noisy nature of lattice-based assumptions. This limit on the number of updates was lifted in [2], by Albrecht et al. but the resulting scheme has limited practicality, as ciphertext and update token sizes are about 12 KiB and 1486 KiB respectively for the NIST level I (128 bits) security level. Moreover, the suggested parameters are vulnerable to the attack of [9].

Concurrently to this work, Patra et al. [46] recently proposed a code-based construction with unbounded updates. While the paper claims that its construction is compact (public keys, ciphertexts and update tokens are 4-5 KiB each), the security level achieved clearly falls below NIST level I. In fact the security proof relies on a high-entropy structure preserving map (which we denote by SPM) family that remains uninstantiated. The paper proposes a candidate for this map family, but it has only $n\varphi(n)$ elements (see [46, Remark 1]), with $n \approx 17,669$ for NIST level I. This means that the construction can be seen as a feasibility result, as it cannot be securely instantiated yet.

A natural alternative for post-quantum UPKE is isogeny-based cryptography. In fact, Eaton et al. [29] first proposed two isogeny-based UPKEs. One from SIDH, now broken [16,40,49], and one from CSIDH. More recently, Jana and Dutta [35] also proposed two isogeny-based UPKEs from SiGamal [42] and SimS [33], which are both CSIDH-based as well. These CSIDH constructions are usually instantiated with a 500-bit prime, which is not believed to offer enough quantum security as selecting appropriate quantum-secure CSIDH parameters is still subject to debate [11,13,19,47]. Nowadays, recent advances [21,22,45] in the evaluation of isogeny group actions could be used to instantiate these isogeny group action constructions with any desired parameter set. Nevertheless, building efficient and compact UPKEs from other isogeny assumptions remains an open problem.

The first step in this direction was done by Duparc et al. [28]. They proposed SILBE, the first secure isogeny-based UPKE that was not based on group actions, leveraging so-called generalized lollipop attacks on MSIDH [32] by Castryck and Vercauteren [17]. However, SILBE is a feasibility result which solved an updatability problem encountered by Eaton et al. [29] in their SIDH-based construction.

We summarize the relevant constructions and their main features in Table 1.

Contribution. In this paper, we propose a more efficient isogeny-based UPKE with unbounded updates that does not rely on isogeny group actions. It is built on two key insights.

Our first ingredient is a generalization of the FESTA [8] public-key encryption (PKE) scheme using dimension-four isogenies. This generalization allows us to relax constraints on the degree of the isogenies used compared to the original FESTA and implies a simpler security proof, relying only on the CIST^2 assumption [8, Problem 8]. Moreover, its public keys are uniformly distributed in the supersingular isogeny graph.

UPKE	Family / Assumption	Number of updates	Formal security proof
[27]	Lattices/ LWE	Bounded	Yes
[34]	Lattices/ LWE	Bounded	Yes
[2]	Lattices/ LWE, PCE	Unbounded	Yes
[46]	Codes/ DQCSDP, SPM	Unbounded	Yes
[29]	Isogenies/ SIDH	Unbounded	No
[29]	Isogenies/ CSIDH	Unbounded	Yes
[28]	Isogenies/ MSIDH	Unbounded	No
[35]	Isogenies/ CSIDH	Unbounded	Yes
This paper	Isogenies/ CIST^2, IND-KLPT	Unbounded	Yes

Table 1: Comparison of the main features of current post-quantum UPKEs. The assumptions in red are broken while the constructions with the assumptions in orange do not meet their expected security level.

Using this new PKE as a basis, we build an update mechanism by randomizing the public-key through a long-enough isogeny walk. To compute the corresponding secret key, in Section 3, we explain how to "shorten" an isogeny walk. More specifically, we introduce ConstrainedIsogeny (Algorithm 1), an algorithm that, given the knowledge of the endomorphism ring of two curves, outputs the description of an isogeny of degree $d \leq 2^a\sqrt{p}$ between them matching some constraints on the 2^a-torsion. Crucially, this prevents the update mechanism from having secret isogenies whose degrees grow with the number of updates. This algorithm is therefore a key ingredient to be able to allow an unbounded amount of key updates.

We provide a formal proof of the security of the UPKE, relying on the CIST^2 assumption [8, Problem 8] and on the IND-KLPT assumption (Assumption 1) which states that outputs of the KLPT algorithm are indistinguishable from random isogenies of the same degree.

Finally, we theoretically compare our construction to SILBE. Our construction is expected to be several orders of magnitude more efficient and is more compact. In fact, as discussed in [28, §5.2], the decryption and the secret key update mechanism in SILBE are expected to cost at least 2^{60} and 2^{57} field operations over $\mathbb{F}_{p^2}$ where $p \sim 2^{13013}$. This is because in SILBE, the decryption and the secret key update mechanisms require the computation of a dimension 4 isogeny of degree N where N is the product of all the primes between 5 and 6863. In contrast, for our UPKE, the base prime p is roughly 1631 bits long, the decryption and the secret key update mechanisms require the computation of a shorter four-dimensional isogeny whose degree is a power of 2. Our construction is hence expected to be several orders of magnitude more efficient. Moreover, the sizes of the public key, the ciphertext and the update token in SILBE are about 3.18 KiB, 7.95 KiB and 7.95 KiB respectively, while they are 0.4 KiB, 2.13 KiB and 2.13 KiB respectively for our construction. While an initial implementation of Encryption and ConstrainedIsogeny (Algorithm 1) is provided, we leave a full implementation of our UPKE for future work.

Acknowledgements. We are grateful to the anonymous reviewers of Asiacrypt 2026 for their suggestions that helped us improve the paper. In particular, we are thankful for their suggestion of implementing ConstrainedIsogeny (Algorithm 1) only relying on quaternion operations.

Basso was partially supported by SNSF Consolidator Grant CryptonIs 21376. Maino and Kutas were partially supported by Engineering and Physical Sciences Research Council (EPSRC) grant number EP/V011324/1. Kouider and Kutas were supported by the Ministry of Culture and Innovation and the National Research, Development, and Innovation Office within the Quantum Information National Laboratory of Hungary (Grant No. 2022-2.1.1-NL-2022-00004). Kutas is also supported by the NRDIO grant "EXCELLENCE-151343".

2 Preliminaries

Notation. Throughout this paper, the quantity λ denotes a security parameter. We assume that all algorithms will have implicit access to it. Following NIST security levels, we have $\lambda \in \{128, 192, 256\}$, which stands for NIST level I, III and V, respectively. We will mostly be working with the finite field $\mathbb{F}_{p^2}$ where $p > 0$ is a prime. Given two integers a and b, $[a, b)$ denotes the set of integers x such that $a \leq x < b$. We denote by $a \,||\, b$ the concatenation of the values a and b. Finally, we use PPT as an abbreviation for probabilistic polynomial time.

2.1 Isogenies

Let E and E' be two elliptic curves defined over some finite field $\mathbb{F}_q$ where $q = p^n$. An *isogeny* $\varphi : E \to E'$ is a surjective rational map which is also a group homomorphism with respect to the elliptic curve group law on E and E'. The kernel of an isogeny φ is always finite and $\#\ker\varphi$ is called the *separable degree* of φ. When the degree of φ (as a rational map) is equal to its separable degree, we say that φ is *separable*. Unless stated otherwise, the isogenies considered in this work are separable. Given an isogeny $\varphi : E \to E'$, there exists a unique (up to isomorphism) *dual isogeny* $\widehat{\varphi} : E' \to E$ such that $\varphi \circ \widehat{\varphi} = [\deg(\varphi)]$ on E' and $\widehat{\varphi} \circ \varphi = [\deg(\varphi)]$ on E.

Efficient isogeny representation. An *efficient representation* of an isogeny $\varphi\colon E \to E'$ is any data of polynomial size that allows one to efficiently evaluate the isogeny φ on any (accessible) point of E. When the degree of φ is smooth and $\ker\varphi$ is accessible, then any representation of generator(s) of $\ker\varphi$ is an efficient representation of φ as one can compute/evaluate φ using the Vélu formulas [50] or the square-root Vélu formulas [10]. When φ is a composition of smooth degree isogenies, the isogeny φ can be represented by concatenating the representations of its factors. When the degree of φ is not smooth, the kernel representation becomes inefficient. Nevertheless, if one knows an evaluation of φ on some accessible torsion points of smooth order $N > \deg\varphi$, then as an application of Kani's lemma [37], Robert [48,49] described an algorithm that embeds φ into a

higher dimensional (2, 4 or 8) isogeny Φ of degree N in such a way that from the evaluation of Φ on torsion points, one can retrieve the evaluation of φ on torsion points. This hence allows the *HD (higher dimensional) representation*, where one represents an isogeny φ using its evaluation on accessible torsion points of smooth order $N > \sqrt{\deg \varphi}$. The requirement $N > \deg \varphi$ can be relaxed to $N > \sqrt{\deg \varphi}$ when working with representations that use dimension 4 or 8 isogenies. This can be very useful when designing schemes, as the representation of the isogeny at play requires much less torsion. We will in fact build our UPKE on a variant of FESTA where one uses dimension-4 representations instead of dimension 2 representations during decryption. In practice N is chosen to be a power of 2. In this work, we manipulate both kernel and HD representations. When giving $\varphi : E \to E'$ as an input or returning it as an output of an algorithm, we intend an efficient representation of φ.

Deuring Correspondence. An isogeny $\varphi : E \to E$ is called an *endomorphism* of E. The set $\mathrm{End}(E)$ of all endomorphisms of E is an integral ring of characteristic zero under addition and composition. In characteristic $p > 0$, $\mathrm{End}(E)$ has rank 2 or 4. In the first case, $\mathrm{End}(E)$ is isomorphic to an order of an imaginary quadratic field, and we say that E is *ordinary*. In the second case, $\mathrm{End}(E)$ is isomorphic to a maximal order $\mathcal{O}$ of $\mathcal{B}_{p,\infty}$, the quaternion algebra ramified at p and ∞, and we say that E is *supersingular*. For example, let $p \equiv 3 \mod 4$ be a prime. Then $\mathcal{B}_{p,\infty} \simeq \mathbb{Q} + \mathbb{Q}i + \mathbb{Q}j + \mathbb{Q}k$ where $i^2 = -1$, $j^2 = -p$, $k = ij = -ji$. The elliptic curve $E_0 : y^2 = x^3 + x$ is supersingular. Its endomorphism ring $\mathrm{End}(E_0)$ is isomorphic to the maximal order $\mathcal{O}_0 = \mathbb{Z} + \mathbb{Z}i + \mathbb{Z}\frac{i+j}{2} + \mathbb{Z}\frac{1+k}{2}$ through the isomorphism from $\mathrm{End}(E_0)$ to $\mathcal{O}_0$ given by $\iota \mapsto i$ and $\pi \mapsto j$ where

$$\iota : (x, y) \mapsto (-x, \sqrt{-1}y) \quad \text{and} \quad \pi : (x, y) \mapsto (x^p, y^p).$$

In fact, there is a correspondence, named Deuring correspondence [26], between supersingular elliptic curves in characteristic p (up to isomorphism and Galois conjugacy) and maximal orders (up to conjugacy) of $\mathcal{B}_{p,\infty}$, which sends a supersingular elliptic curve to its abstract endomorphism ring and isogenies to ideals. Table 2 summarizes the Deuring correspondence.

Table 2: The Deuring correspondence.

Supersingular j-invariants over $\mathbb{F}_{p^2}$	Maximal orders in $\mathcal{B}_{p,\infty}$
$j(E)$ (up to Galois conjugacy)	$\mathcal{O} \cong \mathrm{End}(E)$ (up to conjugacy)
(E_1, φ) with $\varphi : E \to E_1$	I_φ, integral left $\mathcal{O}$-ideal and right $\mathcal{O}_1$-ideal
$\theta \in \mathrm{End}(E)$	Principal ideal $\mathcal{O}\theta$
$\deg(\varphi)$	$n(I_\varphi)$
$\widehat{\varphi}$	$\overline{I_\varphi}$
$\varphi : E \to E_1$ and $\psi : E \to E_1$	$I_\varphi \sim I_\psi$
$\psi \circ \varphi : E \to E_1 \to E_2$	$I_{\psi\circ\varphi} = I_\varphi \cdot I_\psi$

2.2 Known algorithms

For ease of reading, we gather here all the known, isogeny-related algorithms we will use in the rest of the paper.

- $(E', \varphi) \leftarrow \mathsf{IsogenyWalk}_\ell(E, n, r)$: on input a curve E, an integer n and $r = (r_0, r_1, \cdots, r_{n-1}) \in [0, \ell+1) \times [0, \ell)^{n-1}$, the algorithm returns (E', φ), where $\varphi : E \to E'$ is the non-backtracking isogeny walk of length n defined by r in the ℓ-isogeny graph. The isogeny $\varphi = \varphi_{n-1} \circ \ldots \varphi_0$ is computed step-by-step, in a CGL-like fashion [18]: the kernel of each isogeny φ_i is determined by r_i. When $n \geq \epsilon \log_\ell(p)$, then the distribution of the curve E' is at distance $O(2^{-\frac{1}{2}(\epsilon-1)\log_2(p)})$ from the stationary distribution on the ℓ-isogeny graph [7, Theorem 11].
- $(P, Q) \leftarrow \mathsf{Gen2Tor}(E)$: on input a curve E, it outputs a deterministic basis (P, Q) of $E[2^a]$, where $a = \max\{n : 2^n \mid (p+1)\}$. In the literature, there exist several ways to do so; see for instance [1, §2.2.3] or [51].
- $\mathrm{End}(E') \leftarrow \mathsf{IsogenyToEnd}(\varphi : E \to E', \mathrm{End}(E))$: on input an isogeny $\varphi : E \to E'$ of degree 3^n for some $n \geq 1$, and $\mathrm{End}(E)$, this algorithm returns $\mathrm{End}(E')$. This algorithm is a straightforward generalization of [1, Algorithm 3.17]. We write the isogeny $\varphi = \varphi_r \circ \ldots \circ \varphi_1 : E \to E_1 \to \ldots \to E_r = E'$, where each φ_i has rational kernel generators. Then, each isogeny φ_i is translated to its corresponding ideal I_{φ_i}, where we have that $\mathrm{End}(E_i)$ is the right order of I_{φ_i}. The output is $\mathrm{End}(E_r)$.
- $r \leftarrow \mathsf{KLPT}_\ell(E, \mathrm{End}(E), E', \mathrm{End}(E'))$: on input two supersingular elliptic curves E and E' together with their endomorphism rings $\mathrm{End}(E)$ and $\mathrm{End}(E')$, this algorithm returns $r = (r_0, r_1, \cdots, r_{n-1}) \in [0, \ell+1) \times [0, \ell)^{n-1}$, such that $(E', \varphi) \leftarrow \mathsf{IsogenyWalk}_\ell(E, n, r)$ where $\varphi : E \to E'$ is an isogeny of degree ℓ^n. It runs the KLPT [38] algorithm used in earlier versions of SQIsign [1,25,30,20] to get an isogeny $\varphi : E \to E'$ of degree $\ell^n \approx p^{15/4}$, then recovers the output $r = (r_0, r_1, \cdots, r_{n-1}) \in [0, \ell+1) \times [0, \ell)^{n-1}$ by mapping the kernel of each degree ℓ factor φ_i of φ to the corresponding scalars $r_0 \in [0, \ell+1)$ and $r_i \in [0, \ell)$ for $i = 1, \cdots, n-1$.

In our UPKE, we will be performing random walks of length $\delta = \lceil \frac{15}{4} \log_3(p) \rceil$ with $\ell = 3$. This ensures that (i) the distribution of the end curve is at negligible distance from the stationary distribution, and (ii) given two elliptic curves E and E' of known endomorphism ring, one can efficiently compute an isogeny $\varphi : E \to E'$ of degree 3^δ using KLPT.

2.3 FESTA

In this section, we recall FESTA [8], an IND-CCA public-key encryption mechanism obtained by applying the OAEP transform to a trapdoor function; we summarize the trapdoor function in what follows. We define E_0 to be a supersingular elliptic curve over $\mathbb{F}_{p^2}$, where $p \equiv 3 \pmod 4$ and $2^a | (p+1)$, for some large enough $a > 0$. Additionally, we fix (P_0, Q_0) to be a basis of $E_0[2^a]$.

The key generation works as follows. We compute an isogeny $\varphi_{\mathsf{sk}} : E_0 \to E_{\mathsf{pk}}$ and then define $(R_{\mathsf{pk}}, S_{\mathsf{pk}}) = ([\alpha_{\mathrm{sk}}]\varphi_{\mathsf{sk}}(P_0), [\beta_{\mathrm{sk}}]\varphi_{\mathsf{sk}}(Q_0))$, for some odd $\alpha_{\mathrm{sk}}, \beta_{\mathrm{sk}}$. The secret key sk is equal to $(\varphi_{\mathsf{sk}}, \alpha_{\mathrm{sk}}, \beta_{\mathrm{sk}})$, whereas the public key pk consists in $(E_{\mathsf{pk}}, R_{\mathsf{pk}}, S_{\mathsf{pk}})$.

In the evaluation phase, the trapdoor function takes as input two isogenies $\varphi_{\mathsf{Enc}_1} : E_0 \to E_1$ and $\varphi_{\mathsf{Enc}_2} : E_{\mathsf{pk}} \to E_2$ such that $\deg(\varphi_{\mathsf{sk}})$, $\deg(\varphi_{\mathsf{Enc}_1})$ and $\deg(\varphi_{\mathsf{Enc}_2})$ are pairwise coprime. The output is the tuple

$$(E_1, R_1, S_1, E_2, R_2, S_2),$$

where

$$(R_1, S_1) = ([\alpha_{\mathrm{eval}}]\varphi_{\mathsf{Enc}_1}(P_0), [\beta_{\mathrm{eval}}]\varphi_{\mathsf{Enc}_1}(Q_0)) \quad \text{and}$$
$$(R_2, S_2) = ([\alpha_{\mathrm{eval}}]\varphi_{\mathsf{Enc}_2}(R_{\mathsf{pk}}), [\beta_{\mathrm{eval}}]\varphi_{\mathsf{Enc}_2}(S_{\mathsf{pk}})).$$

Now, let $\varphi_{\mathsf{Dec}} := \varphi_{\mathsf{Enc}_2} \circ \varphi_{\mathsf{sk}} \circ \widehat{\varphi}_{\mathsf{Enc}_1} : E_1 \to E_2$. To invert the trapdoor function, we observe that

$$\varphi_{\mathsf{Dec}} \begin{pmatrix} R_1 \\ S_1 \end{pmatrix} = [\deg(\varphi_{\mathsf{Enc}_1})] \begin{pmatrix} [\alpha_{\mathrm{sk}}^{-1}]R_2 \\ [\beta_{\mathrm{sk}}^{-1}]S_2 \end{pmatrix}.$$

This implies that, using the secret key sk, we can recover a higher-dimensional (HD) representation of φ_{Dec}, which in turn allows us to compute φ_{Enc_1} and φ_{Enc_2}. We give a visual description of the FESTA trapdoor function in Figure 1.

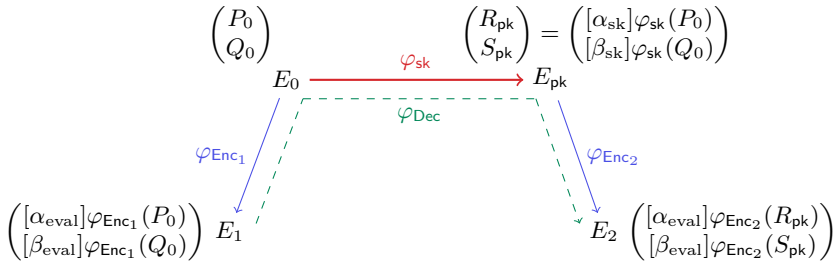

Fig. 1: The FESTA trapdoor function.

2.4 Updatable Public-Key Encryption

In this section, we recall the definition of updatable public-key encryption (UPKE) schemes and associated security notions following the descriptions given in [2,27,35]. One can find a reminder on public-key encryption as well as additional security notions for UPKE in Appendix A.

Definition 1. *An updatable public-key encryption (UPKE) scheme for a message space $\mathcal{M}$ is a tuple* $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{UpdPk}, \mathsf{UpdSk})$ *of PPT algorithms such that:*

- $\mathsf{KeyGen}(\lambda) \to (\mathsf{sk}, \mathsf{pk})$: KeyGen *is a probabilistic key generation algorithm that outputs a secret key* sk *and a public key* pk *for a given security parameter* λ*;*
- $\mathsf{Enc}(\mathsf{pk}, \mathsf{mex}) \to \mathsf{ctxt}$: Enc *takes as input a public key* pk *and a message* $\mathsf{mex} \in \mathcal{M}$*, and outputs a ciphertext* ctxt*;*
- $\mathsf{Dec}(\mathsf{sk}, \mathsf{ctxt}) \to \mathsf{mex}$: Dec *takes as input a secret key* sk *and a ciphertext* ctxt*, and outputs a message* mex*;*
- $\mathsf{UpdPk}(\mathsf{pk}, \rho) \to (\mathsf{pk}', \mathsf{up})$: UpdPk *takes as input a public key* pk *and randomness* $\rho \in \mathcal{R}$*, and produces an updated public key* pk' *along with an update token* up*;*
- $\mathsf{UpdSk}(\mathsf{sk}, \mathsf{up}) \to \mathsf{sk}'$: UpdSk *takes as input a secret key* sk *together with an update token* up*, and outputs an updated secret key* sk'*.*

The various security notions for updatable public-key encryption, as established in [27], follow closely the hierarchy established for the security of public-key encryption, but they include an additional key update oracle. Namely, we distinguish one-way (OW) security, indistinguishability under chosen-plaintext (IND-CPA) security and indistinguishability under chosen ciphertext (IND-CCA) security. Furthermore, each of these notions comes with two additional variants, which depend on the modeling of the key update oracle, namely chosen randomness (CR) where the adversary controls only the randomness used by the update oracle and chosen updates (CU), where the adversary can create entirely malicious public keys and update tokens.

Note that IND-CR-CPA is the standard notion that is usually proven for post-quantum updatable public key encryption schemes, such as in the constructions [35] and [2]. Several works have shown generic transforms from IND-CR-CPA to IND-CU-CCA security, or from OW-CR-CPA to IND-CU-CCA, such as [27,6]. This stronger security notion is therefore achievable from any IND-CR-CPA or OW-CR-CPA secure scheme, by using the generic transforms. In this work, we will use the transform introduced by [6] to turn an OW-CR-CPA secure UPKE into an IND-CU-CCA secure one.

We define below the notion of OW-CR-CPA secure UPKE.

Definition 2 (OW-CR-CPA security). *The OW-CR-CPA security of a UPKE scheme is defined by a game between an adversary* $\mathcal{A}$ *and a challenger* $\mathcal{C}$ *defined in Figure 2.* $\mathcal{A}$*'s advantage is defined by:*

$$\mathsf{Adv}_{\Pi}^{\mathsf{OW\text{-}CR\text{-}CPA}}(\mathcal{A}) = \Pr[\mathsf{OW\text{-}CR\text{-}CPA}_{\Pi}^{\mathcal{A}} = 1]. \tag{1}$$

A UPKE scheme is OW-CR-CPA secure if for all PPT *adversaries* $\mathcal{A}$*, its advantage* $\mathsf{Adv}_{\Pi}^{\mathsf{OW\text{-}CR\text{-}CPA}}(\mathcal{A})$ *is negligible.*

In Appendix A, we further formalize an additional property needed for the transform to be valid, namely γ-*spreadness* (Definition 21), and we also explicitly recall the security notion of IND-CU-CCA security (Definition 22).

3 Computing isogenies matching constraints

In this section, we explain how to compute special isogenies between elliptic curves of known endomorphism rings. Such isogenies must act in a prescribed

$\mathsf{OW\text{-}CR\text{-}CPA}^{\mathcal{A}}_{\Pi}(1^\lambda)$	$\mathsf{UpdO}(\rho)$
$i \leftarrow 0$ // key update counter	$(\mathsf{pk}_{i+1}, \mathsf{up}_{i+1}) \leftarrow \mathsf{UpdPk}(\mathsf{pk}_i, \rho)$
$(\mathsf{pk}_0, \mathsf{sk}_0) \leftarrow \mathsf{KeyGen}(1^\lambda)$	$\mathsf{sk}_{i+1} \leftarrow \mathsf{UpdSk}(\mathsf{sk}_i, \mathsf{up}_{i+1})$
$\mathsf{state} \leftarrow \mathcal{A}^{\mathsf{UpdO}(\rho)}(\mathsf{pk}_0)$	$i \leftarrow i+1$
$\mathsf{q} \leftarrow i$	
$\mathsf{mex}^* \leftarrow_\$ \mathcal{M}$	
$\mathsf{ctxt}^* \leftarrow \mathsf{Enc}(\mathsf{pk}_\mathsf{q}, \mathsf{mex}^*)$	
$\mathsf{state} \leftarrow \mathcal{A}^{\mathsf{UpdO}(\rho)}(\mathsf{ctxt}^*, \mathsf{state})$	
$\mathsf{q}' \leftarrow i$	
$\mathsf{r}^* \leftarrow_\$ \mathcal{R}$	
$(\mathsf{pk}^*, \mathsf{up}^*) \leftarrow \mathsf{UpdPk}(\mathsf{pk}_{\mathsf{q}'}, \mathsf{r}^*)$	
$\mathsf{sk}^* \leftarrow \mathsf{UpdSk}(\mathsf{sk}_{\mathsf{q}'}, \mathsf{up}^*)$	
$\mathsf{mex}' \leftarrow \mathcal{A}(\mathsf{ctxt}^*, \mathsf{pk}^*, \mathsf{sk}^*, \mathsf{up}^*, \mathsf{state})$	
return $(\mathsf{mex}' = \mathsf{mex}^*)$	

Fig. 2: OW-CR-CPA game for a UPKE scheme Π.

way on some torsion. Before specializing to our use case, we provide some simple structural results.

It is well-known that isogenies between two supersingular elliptic curves form a 4-dimensional lattice. Let N be an integer coprime to p and let E_1, E_2 be supersingular elliptic curves. Also, fix two bases (P_1, Q_1) and (P_2, Q_2) of the N-torsion of E_1 and E_2, respectively. We now introduce some notation that we will use to define a sublattice of $\mathrm{Hom}(E_1, E_2)$.

Definition 3. *Let E_1, E_2 be supersingular elliptic curves and let P_1, Q_1 and P_2, Q_2 be fixed bases of the N-torsion of E_1 and E_2, respectively. We denote by $\mathrm{Hom}(E_1, E_2)_{(P_1,Q_1),(P_2,Q_2)}$ the set of isogenies taking P_1 to the subgroup generated by P_2 and taking Q_1 to the subgroup generated by Q_2 , i.e.,*

$$\mathrm{Hom}(E_1, E_2)_{(P_1,Q_1),(P_2,Q_2)} \coloneqq \left\{ \varphi \in \mathrm{Hom}(E_1, E_2) \;\middle|\; \begin{array}{l} \varphi(\langle P_1 \rangle) \leqslant \langle P_2 \rangle, \\ \varphi(\langle Q_1 \rangle) \leqslant \langle Q_2 \rangle \end{array} \right\}.$$

Our goal is to prove that $\mathrm{Hom}(E_1, E_2)_{(P_1,Q_1),(P_2,Q_2)}$ is a 4-dimensional lattice. Before that, we recall that endomorphisms that fix the subgroup generated by a point P are of the form $\mathbb{Z} + I_\varphi$, where φ is the isogeny whose kernel is generated by P and I_φ is the corresponding left ideal. The set $\mathbb{Z} + I_\varphi$ is also the Eichler order obtained from intersecting $\mathrm{End}(E_1)$ with the right order of I_φ. We will call this the *Eichler order associated to P*.

Proposition 4. *Let E_1, E_2 be supersingular elliptic curves and let (P_1, Q_1) and (P_2, Q_2) be fixed bases of the N-torsion of E_1 and E_2 respectively. Then the following hold:*

1. $\mathrm{Hom}(E_1, E_2)_{(P_1,Q_1),(P_2,Q_2)}$ *is not empty.*
2. *Let* $\mathcal{I}_{P_1}$ *and* $\mathcal{I}_{Q_1}$ *be the Eichler orders associated to* P_1 *and* Q_1*. Then* $\mathrm{Hom}(E_1, E_2)_{(P_1,Q_1),(P_2,Q_2)}$ *is a right* $\mathcal{I}_{P_1} \cap \mathcal{I}_{Q_1}$*-module.*
3. *As a right* $\mathcal{I}_{P_1} \cap \mathcal{I}_{Q_1}$*-module,* $\mathrm{Hom}(E_1, E_2)_{(P_1,Q_1),(P_2,Q_2)}$ *is isomorphic to a right ideal in* $\mathcal{I}_{P_1} \cap \mathcal{I}_{Q_1}$*.*

Proof. [31, Lemma 3.4] implies that there exist isogenies for any prescribed action on the N-torsion, thus $\mathrm{Hom}(E_1, E_2)_{(P_1,Q_1),(P_2,Q_2)}$ is not empty.

The second claim follows from the fact that elements of $\mathcal{I}_{P_1} \cap \mathcal{I}_{Q_1}$ fix both the subgroup generated by P_1 and Q_1. Indeed, $\mathcal{I}_{P_1}$ consists of endomorphisms of the form $\mathbb{Z} + I_{P_1}$, where $I_{P_1} = \{\alpha \in \mathrm{End}(E_1) \mid \alpha(P_1) = 0\}$; hence the claim that elements of $\mathbb{Z} + I_{P_1}$ send P_1 to an integer multiple of P_1. The same holds for $\mathcal{I}_{Q_1}$ and Q_1.

As for the third claim, we show a concrete isomorphism. We take an isogeny $\psi : E_2 \to E_1$ that sends the subgroup generated by P_2 to the subgroup generated by P_1 and the subgroup generated by Q_2 to the subgroup generated by Q_1; this exists because of [31, Lemma 3.4]. Then, composing ψ with $\mathrm{Hom}(E_1, E_2)_{(P_1,Q_1),(P_2,Q_2)}$ yields a set J that is a collection of endomorphisms in $\mathcal{I}_{P_1} \cap \mathcal{I}_{Q_1}$. Furthermore, since $\mathrm{Hom}(E_1, E_2)_{(P_1,Q_1),(P_2,Q_2)}$ is a right $\mathcal{I}_{P_1} \cap \mathcal{I}_{Q_1}$-module, J is a right ideal in $\mathcal{I}_{P_1} \cap \mathcal{I}_{Q_1}$. □

Proposition 4 implies that $\mathrm{Hom}(E_1, E_2)_{(P_1,Q_1),(P_2,Q_2)}$ is a 4-dimensional lattice. We now prove that the discriminant of $\mathrm{Hom}(E_1, E_2)_{(P_1,Q_1),(P_2,Q_2)}$ is equal to p^2M^2, where M is a divisor of N^2. First, we recall the following easy result.

Lemma 5. *Let* G *be an abelian group and let* H, K *be subgroups of finite index in* G*. Then* $H \cap K$ *is also of finite index and* $|G : (H \cap K)|$ *divides* $|G : H||G : K|$*.*

Proof. We define the following homomorphism of groups: $\theta : G/(H \cap K) \to G/H \times G/K, \theta(g) = (g \mod H, g \mod K)$. We show that θ is injective. Assume that $g_1 \equiv g_2 \pmod{H}$ and $g_1 \equiv g_2 \pmod{K}$. This implies that $g_1 g_2^{-1}$ is both in H and K, hence it lies in $H \cap K$ which proves injectivity. Now the statement follows from the first isomorphism theorem (or even Lagrange's theorem). □

To prove that the determinant of $\mathrm{Hom}(E_1, E_2)_{(P_1,Q_1),(P_2,Q_2)}$ is equal to p^2M^2 where M is a divisor of N^2, we observe what follows. Let $\mathcal{O}_1$, $\mathcal{O}_2$ and $\mathcal{O}_3$ be three maximal orders where $\mathcal{O}_1 \cap \mathcal{O}_2$ and $\mathcal{O}_1 \cap \mathcal{O}_3$ are Eichler orders of level N; that is $\mathcal{O}_1 \cap \mathcal{O}_2$ and $\mathcal{O}_1 \cap \mathcal{O}_3$ are both additive subgroups of $\mathcal{O}_1$ of index N. By Lemma 5, we have that the index of $\mathcal{O}_1 \cap \mathcal{O}_2 \cap \mathcal{O}_3$ in $\mathcal{O}_1$ is equal to M, a divisor of N^2. Now the upper bound on the discriminant p^2N^4 follows from the fact that a sublattice of index M of an order $\mathcal{O}$ has discriminant p^2M^2.

3.1 Specializing to our use case

Equipped with the above theoretical tools, we now describe an algorithm to sample isogenies from $\mathrm{Hom}(E_1, E_2)_{(P_1,Q_1),(P_2,Q_2)}$ of small norm. For our purposes we will assume that we have a prime of the form $p = f2^a3^b5 - 1$, where f is a

small cofactor and $2^a > 3^b 5^c \sqrt{p}(2a)^3$, where c is such that $5^c \approx 3^b$; this technical condition will become clear in what follows. We also fix $E_0 : y^2 = x^3 + x$, whose endomorphism ring $\mathrm{End}(E_0)$ is given by all the integral combinations of the endomorphisms $[1]$, ι, $\frac{\iota+\pi}{2}$ and $\frac{1+\iota\pi}{2}$ (see Section 2.1), as well as (P_0, Q_0), which is the output of $\mathsf{Gen2Tor}(E_0)$.

We assume that we are given an elliptic curve E together with its endomorphism ring $\mathrm{End}(E)$. Let $(P, Q) \leftarrow \mathsf{Gen2Tor}(E)$. Our goal is to compute an isogeny representation of an isogeny $\varphi : E_0 \to E$ such that $(\varphi(P_0), \varphi(Q_0)) = ([c_P]P, [c_Q]Q)$ for some $c_P, c_Q \in \mathbb{Z}/N\mathbb{Z}$. To represent the isogeny φ, we will use the information coming from the evaluation of the 2^a-torsion. Similarly to what is done in FESTA, and as we will explain in Section 4.4, in our construction, we will need to provide a representation for an isogeny $\varphi_2 \circ \varphi \circ \widehat{\varphi}_1$, where $\deg(\varphi_1) = 3^b$ and $\deg(\varphi_2) = 5^c$. This translates to the condition $2^{2a} > 3^b 5^c \deg(\varphi)$.

We remark that the isogeny φ we are after is an isogeny in the lattice $\mathrm{Hom}(E_0, E)_{(P_0,Q_0),(P,Q)}$. As we have already shown in the previous section, the discriminant of $\mathrm{Hom}(E_0, E)_{(P_0,Q_0),(P,Q)}$ is upper bounded by $p^2 2^{4a}$. This means that the shortest isogeny (i.e., the isogeny of smallest degree) in the lattice $\mathrm{Hom}(E_0, E)_{(P_0,Q_0),(P,Q)}$ will have degree $\approx \sqrt{p}2^a$. Hence, to represent $\varphi_2 \circ \varphi \circ \widehat{\varphi}_1$, we need $2^{2a} > 3^b 5^c \deg(\varphi) \approx \sqrt{p}2^a 3^b 5^c$, which is satisfied since we imposed $2^a > 3^b 5^c \sqrt{p}(2a)^3$. We next explain how to sample such a short isogeny in $\mathrm{Hom}(E_0, E)_{(P_0,Q_0),(P,Q)}$.

While there exist polynomial-time algorithms to compute Hom lattices between supersingular elliptic curves, in general implementing them would provide a substantial overhead for our UPKE scheme. Instead we take an approach that solves finding the above constrained isogeny purely on the quaternion side.[7]

We assume that we have as input an isogeny of odd degree $\varphi : E_0 \to E$ together with the corresponding left ideal I_φ (this aligns with the setup of our UPKE scheme).

First one takes an endomorphism α on E_0 such that $\alpha(P_0) = \widehat{\varphi}(P)$ and $\alpha(Q_0) = \widehat{\varphi}(Q)$. Since we know the endomorphism ring of E_0, we can compute the action of the basis on the 2^a-torsion and then, after computing some discrete logarithms and solving a linear system, we can compute a suitable endomorphism α. Now, let I be the left ideal of E_0 corresponding to the isogeny $\varphi \circ \alpha$. Geometrically, the ideal I consists of endomorphisms that can be decomposed as $\psi \circ \varphi \circ \alpha$ for some $\psi \in \mathrm{Hom}(E, E_0)$. Now let us examine the set $I \cap \Lambda_{P_0} \cap \Lambda_{Q_0}$. This set consists of isogenies that are of the form $\psi \circ \varphi \circ \alpha$ and map P_0 to the subgroup generated by P_0 and map Q_0 to the subgroup generated by Q_0. Now one has that: $\psi \circ \varphi \circ \alpha \begin{pmatrix} P_0 \\ Q_0 \end{pmatrix} = \psi \circ \varphi \circ \widehat{\varphi} \begin{pmatrix} P \\ Q \end{pmatrix} = [\deg(\varphi)] \circ \psi \begin{pmatrix} P \\ Q \end{pmatrix}$.

The above equality implies that $\widehat{\psi}$ lies in $\mathrm{Hom}(E_0, E)_{(P_0,Q_0),(P,Q)}$. Furthermore, the above calculation also shows that for any ψ such that $\widehat{\psi}$ lies in $\mathrm{Hom}(E_0, E)_{(P_0,Q_0),(P,Q)}$, the isogeny $\psi \circ \varphi \circ \alpha$ will lie in $I \cap \Lambda_{P_0} \cap \Lambda_{Q_0}$. Now, our goal is to find short elements (with respect to quaternion norms) satisfying

[7] We note that this approach was sketched to us by an anonymous reviewer, for which we are very grateful.

certain constraints (coming from efficient HD representations) which we will recall shortly. Before we move on to the non-size-related restrictions we want to emphasize that once a suitable element $\beta \in I \cap \Lambda_{P_0} \cap \Lambda_{Q_0}$ is found we do not need to run any ideal-to-isogeny subroutines. Namely, for decryption it is actually enough to know what $\widehat{\psi}(P_0)$ and $\widehat{\psi}(Q_0)$ are.

This can be computed as follows. We have β as an explicit endomorphism of E_0, hence we know what $\beta(P_0)$ and $\beta(Q_0)$ are. Recall that β decomposes as $\beta = \psi \circ \varphi \circ \alpha$; as a result, we have that $\alpha(P_0) = \hat{\varphi}(P)$, which implies that $\beta(P_0) = [\deg(\varphi)]\psi(P)$ and we also have that $\beta(Q_0) = [\deg(\varphi)]\psi(Q)$. Since β is known and $\deg(\varphi)$ is known, one can compute $\psi(P)$ and $\psi(Q)$ from which $\widehat{\psi}(P_0)$ and $\widehat{\psi}(Q_0)$ can be computed efficiently (assuming that the degree of ψ is odd).

Now we turn our attention to additional constraints on β coming from HD representation concerns. As we mentioned earlier, in our construction, we will need to represent $\varphi_2 \circ \varphi \circ \widehat{\varphi}_1 : E_1 \to E_2$ for some isogenies φ_1 and φ_2 such that $\deg(\varphi_1) = 3^b$ and $\deg(\varphi_2) = 5^c$. In practice, this means that we need $2^{2a} - d3^b5^c$ to be a sum of two squares; this will allow us to represent the isogeny in dimension four. To have a sum of two squares, it would be desirable that $2^{2a} - d3^b5^c = 1 \bmod 4$. Additionally, for computational reasons, we need that $2^{2a} - d3^b5^c$ is a prime. Clearly, this cannot happen if d is even, so we first need to remove the even part of d.

Define n to be the largest integer such that 2^n divides d and call $d' = d/2^n$. Now, we need to check that $2^{2a-n} - d'3^b5^c = 1 \bmod 4$. If $d'3^b5^c = 1 \bmod 4$, it is not possible that $2^{2a-n} - d'3^b5^c = 1 \bmod 4$. If this is the case, we need to compute an additional 7-isogeny $\chi : E_2 \to E_2'$, so we need to represent the isogeny $\chi \circ \varphi_2 \circ \varphi \circ \widehat{\varphi}_1 : E_1 \to E_2 \to E_2'$. In particular, we will have that $2^{2a-n} - d'3^b5^c7 = 1 \bmod 4$. If $2^{2a-n} - (d/2^n)3^b5^c = 3 \bmod 4$, we will redefine d' to be equal to $7d/2^n$.

It can still happen that $2^{2a-n} - d'3^b5^c$ is not a prime. If this is the case, we check for increasing odd s when $2^{2a-n} - d'3^b5^cs^2$ is a prime. From the isogeny perspective, this means that we need to compose the isogeny we need to represent with a scalar multiplication by s. This does not incur any additional cost.

We assume that the quantity $2^{2a-n} - d'3^b5^cs^2$ behaves as a random integer, so we need to be able to try $\approx 2a$ times before we find a prime; this is due to the prime number theorem. Since we are only trying to add squares to the quantity $d'3^b5^cs^2$, we need that $2^{2a-n} > d'3^b5^c(2a)^2$. Since we do not have control on the exponent n and the fact that we may need an additional factor 7, we will assume that $2^{2a} > 2^a3^b5^c\sqrt{p}(2a)^3$.

We collect what we have described so far in Algorithm 1, which we will denote as ConstrainedIsogeny in the rest of the paper.

Remark 6. Concurrent work [14] has proposed and implemented similar techniques to compute isogenies with torsion constraints, including the constraints from Definition 3. Nevertheless, the concurrent work targets different applications than a UPKE.

Algorithm 1 $\mathsf{ConstrainedIsogeny}(E, \mathrm{End}(E), \varphi, I_\varphi)$

Require: An elliptic curve E together with its endomorphism ring $\mathrm{End}(E)$ and an isogeny representation $\varphi : E_0 \to E$ of odd degree and corresponding left ideal I_φ.

Ensure: A tuple (c_P, c_Q, d, s) describing an isogeny $\psi : E_0 \to E$ such that $\deg(\psi) = d$, $([c_P]P, [c_Q]Q) = (\psi(P_0), \psi(Q_0))$, the quantity $2^{2a-n} - d'3^b5^cs^2$ is prime $= 1 \bmod 4$, where $d' = \begin{cases} d/2^n & \text{if } ((d/2^n)3^b5^c) = 3 \bmod 4 \\ 7d/2^n & \text{otherwise} \end{cases}$ and n is the largest integer such that 2^n divides d, and $(P, Q) \leftarrow \mathsf{Gen2Tor}(E)$.

1: Find $\alpha \in \mathrm{End}(E_0)$ such that $\alpha(P_0) = \widehat{\varphi}(P)$ and $\alpha(Q_0) = \widehat{\varphi}(Q)$
2: Set $I = I_\varphi \alpha$ and compute $\Lambda = I \cap \Lambda_{P_0} \cap \Lambda_{Q_0}$
3: Find the element β of the smallest quaternion norm of Λ using lattice reduction. Let ψ be the isogeny corresponding to β and evaluate ψ on the 2^a-torsion.
4: Compute $c_P, c_Q \in \mathbb{Z}/2^a\mathbb{Z}$ such that $([c_P]P, [c_Q]Q) = (\psi(P_0), \psi(Q_0))$.
5: $d \leftarrow \deg(\psi)$.
6: Compute the largest n such that $2^n \mid d$.
7: **if** $(d/2^n)3^b5^c = 3 \bmod 4$ **then**
8: $\quad d' \leftarrow d/2^n$.
9: **else**
10: $\quad d' \leftarrow 7d/2^n$.
11: Find a small s such that $2^{2a-n} - d'3^b5^cs^2$ is prime.
12: **return** (c_P, c_Q, d, s).

4 Updatable FESTA

In this section, we describe a variant of FESTA we will use to introduce an Updatable public-key encryption. While our variant builds on top of the four-dimensional version of FESTA briefly described in [8, §6.3] together with the PKE construction in [43, §3.2], its main novelty is the fact that every supersingular elliptic curve can be turned into a FESTA public key under our variant. This implies that our public keys are uniformly distributed in the supersingular graph, can be updated with random walks and the distribution before and after the updates are identical. This is crucial in the security proof of our Updatable public-key encryption.

4.1 Parameters

We now fix the parameters we will use in the rest of the paper. Our base prime p is of the form $p = f \cdot 2^a \cdot 3^b \cdot 5 - 1$. With respect to the notations in Section 2.3, $3^b \approx 2^{2\lambda}$ ($b := \lceil 2\lambda / \log_2(3) \rceil$) will be the degree of φ_{Enc_1} and $5^c \approx 2^{2\lambda}$ ($c := \lceil 2\lambda / \log_2(5) \rceil$) will be the degree of φ_{Enc_2}. The (long) secret isogeny and the update isogeny will both have degree 3^δ where $\delta := \left\lceil \frac{15}{4} \log_3(p) \right\rceil$. There are two motivations for this choice of δ. The first is that the public keys are uniformly distributed in the supersingular isogeny graph (see Section 2.2). The second, which will be useful in our security proof, is that given two elliptic curves with known endomorphism rings, one can link them by an isogeny of such degree using KLPT (see Section 2.2).

As in FESTA, during decryption, we will have to compute the isogeny $\varphi_{\mathsf{Dec}} := \varphi_{\mathsf{Enc}_2} \circ \varphi_{\mathsf{sk}} \circ \widehat{\varphi}_{\mathsf{Enc}_1} : E_1 \to E_2$. We will use the 2^a-torsion to represent φ_{Dec}. Nevertheless, this time we will go for a dimension 4 representation which only requires $2^{2a} > \deg(\varphi_{\mathsf{Dec}})$, as opposed to the dimension 2 representation which requires $2^a > \deg(\varphi_{\mathsf{Dec}})$. Combining this with the explanation in Section 3.1, we obtain

$$2^{2a} > \deg(\varphi_{\mathsf{Dec}})(2a)^3 \approx 2^a 3^b 5^c \sqrt{p}(2a)^3 \approx 2^{3a/2} 2^{5\lambda} (2a)^3,$$

Hence, we require $a - 6\log(2a) > 10\lambda$. We will make use of a hash function $\mathsf{H}_{\mathsf{up}} : \{0,1\}^* \to [0,4) \times [0,3)^{\delta-1}$ modeled as a random oracle.

The starting curve is the supersingular elliptic curve $E_0 : y^2 = x^3 + x$ whose endomorphism ring is given by all the integral combinations of the endomorphisms $[1]$, ι, $\frac{\iota+\pi}{2}$ and $\frac{1+\iota\pi}{2}$ (see Section 2.1). We also fix (P_0, Q_0) to be the output of $\mathsf{Gen2Tor}(E_0)$ and (R_3, S_3) to be a basis[8] of $E_0[3^b]$.

Remark 7. From the previous estimates on the exponent a, we have that $p > 2^{12\lambda}$. We can now have a more precise estimate of the statistical distance between the distribution of the public keys in our UPKE and the uniform distribution on the supersingular isogeny graph. From Section 2.2, this distance is roughly $2^{-\frac{1}{2}(15/4-1)\log_2(p)} < 2^{-33/2\lambda} = 2^{-16.5\lambda}$. While this is unnecessarily small with respect to the public-key distribution, it is useful in our security proof.

4.2 Key Generation

The first step in KeyGen is to generate a long isogeny $\widetilde{\varphi}_{\mathsf{sk}} : E_0 \to E_{\mathsf{pk}}$ originating from E_0; such an isogeny is generated by uniformly sampling ρ_{sk} from $\{0,1\}^{2\lambda}$ and then applying $\mathsf{IsogenyWalk}_3(E_0, \delta, \mathsf{H}_{\mathsf{up}}(\rho_{\mathsf{sk}}))$. It is worth noting that, given the length of $r_{\mathsf{sk}} = \mathsf{H}_{\mathsf{up}}(\rho_{\mathsf{sk}})$, the curve E_{pk} is uniformly distributed in the supersingular isogeny graph.

Once the isogeny $\widetilde{\varphi}_{\mathsf{sk}} : E_0 \to E_{\mathsf{pk}}$ has been computed, we retrieve $\mathrm{End}(E_{\mathsf{pk}})$ via IsogenyToEnd. Leveraging the knowledge of $\mathrm{End}(E_{\mathsf{pk}})$, we compute the shorter secret isogeny $\varphi_{\mathsf{sk}} : E_0 \to E_{\mathsf{pk}}$ by applying ConstrainedIsogeny. In particular, we obtain $(\alpha_{\mathsf{sk}}, \beta_{\mathsf{sk}}, d_{\mathsf{sk}}, s_{\mathsf{sk}})$, where $d_{\mathsf{sk}} = \deg(\varphi_{\mathsf{sk}})$, $([\alpha_{\mathsf{sk}}]P_{\mathsf{pk}}, [\beta_{\mathsf{sk}}]Q_{\mathsf{pk}}) = (\varphi_{\mathsf{sk}}(P_0), \varphi_{\mathsf{sk}}(Q_0))$ and $(P_{\mathsf{pk}}, Q_{\mathsf{pk}}) \leftarrow \mathsf{Gen2Tor}(E_{\mathsf{pk}})$.

The public key pk is set to be E_{pk}, whereas the secret sk is set to be $(\alpha_{\mathsf{sk}}, \beta_{\mathsf{sk}}, d_{\mathsf{sk}}, s_{\mathsf{sk}}, \mathrm{End}(E_{\mathsf{pk}}))$. We summarize the procedure in Algorithm 2.

4.3 Encryption

Following the PKE construction in [43, §3.2], we encode messages in the scalars used to mask torsion information: we introduce the algorithm MsgToScalars that converts messages into scalars $\pmod{2^a}$. A message $\mathsf{mex} \in \mathcal{M} = [0, 2^{a-2})$

[8] The basis (P_0, Q_0) must not satisfy the conditions described in [17], i.e., there should not exist an endomorphism $\theta \in \mathrm{End}(E_0)$ of small norm such that either P_0 or Q_0 is an eigenvector for θ. As explained in [17, §5.3], this happens with negligible probability when Gen2Tor is used.

Algorithm 2 KeyGen

Require: The Setup information described in Section 4.1.
Ensure: The public key $\mathsf{pk} = E_{\mathsf{pk}}$ and the secret key $\mathsf{sk} = (\alpha_{\mathsf{sk}}, \beta_{\mathsf{sk}}, d_{\mathsf{sk}}, s_{\mathsf{sk}}, \mathrm{End}(E_{\mathsf{pk}}))$.

$\rho_{\mathsf{sk}} \leftarrow_{\$} \{0,1\}^{2\lambda}$.
$r_{\mathsf{sk}} \leftarrow \mathsf{H}_{\mathsf{up}}(\rho_{\mathsf{sk}})$.
$E_{\mathsf{pk}}, \widetilde{\varphi}_{\mathsf{sk}} \leftarrow \mathsf{IsogenyWalk}_3(E_0, \delta, r_{\mathsf{sk}})$.
$\mathrm{End}(E_{\mathsf{pk}}) \leftarrow \mathsf{IsogenyToEnd}(\mathrm{End}(E_0), \widetilde{\varphi}_{\mathsf{sk}})$.
From the representation of $\mathrm{End}(E_{\mathsf{pk}})$ get $I_{\widetilde{\varphi}_{\mathsf{sk}}}$.
$(\alpha_{\mathsf{sk}}, \beta_{\mathsf{sk}}, d_{\mathsf{sk}}, s_{\mathsf{sk}}) \leftarrow \mathsf{ConstrainedIsogeny}(E_{\mathsf{pk}}, \mathrm{End}(E_{\mathsf{pk}}), \widetilde{\varphi}_{\mathsf{sk}}, I_{\widetilde{\varphi}_{\mathsf{sk}}})$.
$\mathsf{pk} \leftarrow E_{\mathsf{pk}}$.
$\mathsf{sk} \leftarrow (\alpha_{\mathsf{sk}}, \beta_{\mathsf{sk}}, d_{\mathsf{sk}}, s_{\mathsf{sk}}, \mathrm{End}(E_{\mathsf{pk}}))$.
return pk, sk.

is encoded into an odd scalar $\alpha = 2 \cdot \mathsf{mex} + 1$. On input mex, the function MsgToScalars returns (α, β), where $\alpha = 2 \cdot \mathsf{mex} + 1$ and $\beta = \alpha^{-1} \pmod{2^a}$.

Similarly to the evaluation procedure in Section 2.3, during encryption we compute two isogenies $\varphi_{\mathsf{Enc}_1} : E_0 \to E_1$ and $\varphi_{\mathsf{Enc}_2} : E_{\mathsf{pk}} \to E_2$; for efficiency reasons $\deg(\varphi_{\mathsf{Enc}_1}) = 3^b$ and $\deg(\varphi_{\mathsf{Enc}_2}) = 5^c$. Since during decryption we will need to recover $\ker(\varphi_{\mathsf{Enc}_1})$, we must have that $E_0[3^b]$ is defined over $\mathbb{F}_{p^2}$.

To compute $\varphi_{\mathsf{Enc}_1} : E_0 \to E_1$, we sample a nonce $r_3 \leftarrow_{\$} [0, 3^b)$ and compute the isogeny with kernel $\langle R_3 + [r_3]S_3 \rangle$, where (R_3, S_3) is the basis of $E_0[3^b]$ we have fixed. We then define $(R_1, S_1) = ([\alpha_{\mathrm{eval}}]\varphi_{\mathsf{Enc}_1}(P_0), [\beta_{\mathrm{eval}}]\varphi_{\mathsf{Enc}_1}(Q_0))$.

Since we do not need to recover $\ker(\varphi_{\mathsf{Enc}_2})$ during decryption, we compute $\varphi_{\mathsf{Enc}_2} : E_{\mathsf{pk}} \to E_2$ as a chain of c 5-isogenies. This is done by sampling r_5 uniformly at random from $[0, 6) \times [0, 5)^{c-1}$ and then applying $\mathsf{IsogenyWalk}_5(E_{\mathsf{pk}}, c, r_5)$. Similarly, we define $(R_2, S_2) = ([\alpha_{\mathrm{eval}}]\varphi_{\mathsf{Enc}_2}(P_{\mathsf{pk}}), [\beta_{\mathrm{eval}}]\varphi_{\mathsf{Enc}_2}(Q_{\mathsf{pk}}))$. The final ciphertext is then given as $\mathsf{ctxt} = (E_1, R_1, S_1, E_2, R_2, S_2)$. We describe Enc in Algorithm 3.

Algorithm 3 Enc(pk, mex)

Require: A message $\mathsf{mex} \in \mathcal{M} = [0, 2^{a-2})$ and a public key $\mathsf{pk} = E_{\mathsf{pk}}$.
Ensure: A ciphertext $\mathsf{ctxt} = (E_1, R_1, S_1, E_2, R_2, S_2)$.

$r_3 \leftarrow_{\$} [0, 3^b)$.
$\alpha_{\mathrm{eval}}, \beta_{\mathrm{eval}} \leftarrow \mathsf{MsgToScalars}(\mathsf{mex})$.
$(P_{\mathsf{pk}}, Q_{\mathsf{pk}}) \leftarrow \mathsf{Gen2Tor}(E_{\mathsf{pk}})$.
Compute the isogeny $\varphi_{\mathsf{Enc}_1} : E_0 \to E_1$ with kernel $\langle R_3 + [r_3]S_3 \rangle$.
$(R_1, S_1) \leftarrow ([\alpha_{\mathrm{eval}}]\varphi_{\mathsf{Enc}_1}(P_0), [\beta_{\mathrm{eval}}]\varphi_{\mathsf{Enc}_1}(Q_0))$.
$r_5 \leftarrow_{\$} [0, 6) \times [0, 5)^{c-1}$.
$E_2, \varphi_{\mathsf{Enc}_2} \leftarrow \mathsf{IsogenyWalk}_5(E_{\mathsf{pk}}, c, r_5)$.
$(R_2, S_2) \leftarrow ([\alpha_{\mathrm{eval}}]\varphi_{\mathsf{Enc}_2}(P_{\mathsf{pk}}), [\beta_{\mathrm{eval}}]\varphi_{\mathsf{Enc}_2}(Q_{\mathsf{pk}}))$.
return $\mathsf{ctxt} = (E_1, R_1, S_1, E_2, R_2, S_2)$.

4.4 Decryption

To decrypt ciphertexts, we first define a decoding function for $\mathsf{MsgToScalars}$; we name it $\mathsf{ScalarsToMsg}$. The algorithm $\mathsf{ScalarsToMsg}$ recovers a message mex from a tuple (α, β). The first step is to check that $0 < \alpha, \beta < 2^a$ and that α and β are odd; if $\alpha > 2^{a-1}$, then the tuple (α, β) is redefined to be equal to $(2^a - \alpha, 2^a - \beta)$. Then, the algorithm $\mathsf{ScalarsToMsg}$ proceeds to check that $\alpha \cdot \beta = 1 \pmod{2^a}$ and outputs $\mathsf{mex} = \frac{\alpha - 1}{2}$; if $\alpha \cdot \beta \neq 1 \pmod{2^a}$, it returns $\perp$.

Remark 8. The algorithm $\mathsf{ScalarsToMsg}$ normalizes the scalars (α, β) so that $\alpha < 2^{a-1}$. This step is necessary because, during decryption, we will be able to recover the encryption isogeny φ_{Enc_1} up to automorphisms on the codomain curve. Unless the codomain of φ_{Enc_1} is the curve with j-invariant $\in \{0, 1728\}$, which happens with negligible probability, the only automorphisms on the codomain curve are $\pm\,\mathrm{id}$.

We are now ready to introduce the decryption algorithm Dec (Algorithm 5). We recall that from a ciphertext $(E_1, R_1, S_1, E_2, R_2, S_2)$, we have the evaluation of the isogeny $\varphi_{\mathsf{Dec}} := \varphi_{\mathsf{Enc}_2} \circ \varphi_{\mathsf{sk}} \circ \widehat{\varphi}_{\mathsf{Enc}_1} : E_1 \to E_2$ on $E_1[2^a]$, namely

$$\varphi_{\mathsf{Dec}} \begin{pmatrix} R_1 \\ S_1 \end{pmatrix} = [3^b] \begin{pmatrix} [\alpha_{\mathrm{sk}}]R_2 \\ [\beta_{\mathrm{sk}}]S_2 \end{pmatrix}.$$

Given the evaluation of φ_{Dec} on $E_1[2^a]$ and a secret $(\alpha_{\mathrm{sk}}, \beta_{\mathrm{sk}}, d_{\mathsf{sk}}, s_{\mathsf{sk}})$, we explain how to extract $\widehat{\varphi}_{\mathsf{Enc}_1} : E_1 \to E_0$ in $\mathsf{Extract3^b}$ (see Algorithm 4). We first factor out the even degree components of φ_{Dec}. Define $e_\alpha := \max\{n \mid 2^n \text{ divides } \alpha\}$ and $e_\beta := \max\{n \mid 2^n \text{ divides } \beta\}$, and take $e := \min\{e_\alpha, e_\beta\}$. This means that $\varphi_{\mathsf{Dec}} = [2^e]\varphi'_{\mathsf{Dec}}$.

If $e_\alpha \neq e_\beta$, this means that $\deg(\varphi'_{\mathsf{Dec}})$ is even. We now try to factor φ'_{Dec} as $\varphi'_{\mathsf{Dec}} = \varphi'_{\mathsf{Dec,even}} \circ \varphi'_{\mathsf{Dec,odd}}$. To do so, we notice the following relation

$$[2^{\max\{e_\alpha, e_\beta\}}]\varphi'_{\mathsf{Dec,odd}} \begin{pmatrix} R_1 \\ S_1 \end{pmatrix} = \widehat{\varphi}'_{\mathsf{Dec,even}} \begin{pmatrix} [\alpha_{\mathrm{sk}}]R_2 \\ [\beta_{\mathrm{sk}}]S_2 \end{pmatrix}.$$

In particular, if $e_\alpha > e_\beta$, we have that $\ker(\widehat{\varphi}'_{\mathsf{Dec,even}}) = \langle [2^e]S_2 \rangle$; otherwise $\ker(\widehat{\varphi}'_{\mathsf{Dec,even}}) = \langle [2^e]R_2 \rangle$. We now have a representation of an odd-degree isogeny, namely the HD representation of $\varphi'_{\mathsf{Dec,odd}}$, this follows from the discussion in Section 3.1.

From the HD representation of $\varphi'_{\mathsf{Dec,odd}}$, we compute n to be the largest power of 3 such that $[3^n]$ divides $\varphi'_{\mathsf{Dec,odd}}$. If $n = 0$, then $\ker(\widehat{\varphi}_{\mathsf{Enc}_1}) = \ker(\varphi_{\mathsf{Dec}})[3^b]$, which directly gives a representation of $\widehat{\varphi}_{\mathsf{Enc}_1}$ since $E_0[3^b]$ is defined over $\mathbb{F}_{p^2}$. Otherwise, from $(\alpha_{\mathrm{sk}}, \beta_{\mathrm{sk}}, d_{\mathsf{sk}})$, we compute $\mu : E_0 \to \widetilde{E}$ such that $\varphi_{\mathsf{sk}} = \widetilde{\mu} \circ \mu$ and $\deg(\mu) = 3^n$. Then $\widehat{\varphi}_{\mathsf{Enc}_1} = \widehat{\mu} \circ \widetilde{\varphi}_1$, where $\ker(\widetilde{\varphi}_1) = \ker(\varphi'_{\mathsf{Dec,odd}})[3^{b-n}]$.

Finally, from $\widehat{\varphi}_{\mathsf{Enc}_1}$ we recover the scalars (α, β) such that

$$(R_1, S_1) = ([\alpha]\varphi_{\mathsf{Enc}_1}(P_0), [\beta]\varphi_{\mathsf{Enc}_1}(Q_0));$$

such scalars encode the encrypted message.

Algorithm 4 $\mathsf{Extract3^b}(\varphi, (\alpha, \beta, d, s))$

Require: An HD representation on $\psi = \varphi_2 \circ \varphi \circ \varphi_1 : E_1 \to E_0 \to E \to E_2$ of degree $3^b 5^c d$, where $\deg(\varphi_1) = 3^b$, $\deg(\varphi) = d$ and $\deg(\varphi_2) = 5^c$, a tuple (α, β, d, s) describing a representation of $\varphi : E_0 \to E$, where $\deg(\varphi) = d$ and s is the auxiliary quantity defined in $\mathsf{ConstrainedIsogeny}$ (Algorithm 1).

Ensure: A representation of $\varphi_1 : E_1 \to E_0$.

1: $(R_1, S_1) \leftarrow \mathsf{Gen2Tor}(E_1)$.
2: $(R_2, S_2) \leftarrow (\psi(R_1), \psi(S_1))$.
3: $e_\alpha, e_\beta \leftarrow \max\{n \mid 2^n \text{ divides } \alpha\}, \max\{n \mid 2^n \text{ divides } \beta\}$.
4: $e \leftarrow \min\{e_\alpha, e_\beta\}$.
5: **if** $e_\alpha > e_\beta$ **then**
6: Compute $\chi : E_2 \to E_2'$ with kernel $\langle [2^e]S_2 \rangle$.
7: **if** $e_\alpha < e_\beta$ **then**
8: Compute $\chi : E_2 \to E_2'$ with kernel $\langle [2^e]R_2 \rangle$.
9: $(R_2, S_2) \leftarrow (\chi(R_2), \chi(S_2))$.
10: Redefine $\psi : E_1 \to E_2'$ as the odd part of ψ. ▷ $\psi = \widehat{\chi} \circ \psi_{\mathsf{odd}}$
11: $n \leftarrow \max\{s \mid [3^s] \text{ divides } \psi\}$.
12: **if** $n = 0$ **then**
13: **return** a representation of the isogeny with kernel $\ker(\psi)[3^b]$.
14: From (α, β, d, s), compute μ such that $\varphi = \widetilde{\mu} \circ \mu : E_0 \to \widetilde{E} \to E$ and $\deg(\mu) = 3^n$.
15: Compute the isogeny $\tau : E_1 \to \widetilde{E}$ with kernel $\ker(\psi)[3^{b-n}]$.
16: **return** A representation of $\widehat{\mu} \circ \tau$.

Algorithm 5 $\mathsf{Dec}(\mathsf{sk}, \mathsf{ctxt})$

Require: A ciphertext $\mathsf{ctxt} = (E_1, R_1, S_1, E_2, R_2, S_2)$ and a secret key $\mathsf{sk} = (\alpha_{\mathsf{sk}}, \beta_{\mathsf{sk}}, d_{\mathsf{sk}}, s_{\mathsf{sk}}, \mathrm{End}(E_{\mathsf{pk}}))$.

Ensure: A message mex.

1: $(\widetilde{R}_2, \widetilde{S}_2) \leftarrow ([3^b \alpha_{\mathsf{sk}}]R_2, [3^b \beta_{\mathsf{sk}}]S_2)$.
2: Using $(R_1, S_1, \widetilde{R}_2, \widetilde{S}_2)$ compute the representation of an isogeny $\psi : E_1 \to E_2$ of degree $d_{\mathsf{sk}} 3^b 5^c$.
3: $\widetilde{\varphi}_1 : E_1 \to E \leftarrow \mathsf{Extract3^b}(\psi, (\alpha_{\mathsf{sk}}, \beta_{\mathsf{sk}}, d_{\mathsf{sk}}, s_{\mathsf{sk}}))$.
4: Check that E is isomorphic to E_0; if not abort.
5: Define $\varphi : E_0 \to E_1$ to be the dual of $\widetilde{\varphi}_1 : E_1 \to E_0$.
6: Compute the scalars α, β such that $[\alpha]\varphi(P_0) = R_1$ and $[\beta]\varphi(Q_0) = S_1$.
7: **return** $\mathsf{ScalarsToMsg}(\alpha, \beta)$.

4.5 Public-key update

To update a public key pk, we simply compute a long isogeny $\varphi_{\mathsf{up}} : E_{\mathsf{pk}} \to E_{\mathsf{pk}'}$; the updated public key is then given by $\mathsf{pk}' = E_{\mathsf{pk}'}$. The degree of φ_{up} is greater than $p \cdot 2^{2\lambda}$, so that the distribution of $E_{\mathsf{pk}'}$ is at negligible distance from the uniform distribution. This is achieved by feeding some randomness $\rho \in \{0,1\}^{2\lambda}$ to UpdPk (Algorithm 6) that is then digested by $\mathsf{H_{up}}$, together with E_{pk}, to obtain an integer $r \in [0, 3^\delta)$, determining the isogeny walk $\mathsf{IsogenyWalk}_3(E_{\mathsf{pk}}, \delta, r)$. Finally, the update token up is obtained by encrypting the randomness ρ using Enc.

Algorithm 6 $\mathsf{UpdPk}(\mathsf{pk}, \rho)$

Require: Current public key $\mathsf{pk} = E_{\mathsf{pk}}$ and the randomness $\rho \in \{0,1\}^{2\lambda}$.
Ensure: Updated public key $\mathsf{pk}' = E_{\mathsf{pk}'}$ and the update token up.
1: $r \leftarrow \mathsf{H}_{\mathsf{up}}(\rho \,||\, E_{\mathsf{pk}})$.
2: $E_{\mathsf{pk}'}, _ \leftarrow \mathsf{IsogenyWalk}_3(E_{\mathsf{pk}}, \delta, r)$.
3: $\mathsf{up} \leftarrow \mathsf{Enc}(\mathsf{pk}, \rho)$.
4: **return** $E_{\mathsf{pk}'}, \mathsf{up}$.

4.6 Secret-key update

Given a secret key $\mathsf{sk} = (\alpha_{\mathrm{sk}}, \beta_{\mathrm{sk}}, d_{\mathsf{sk}}, s_{\mathsf{sk}}, \mathrm{End}(E_{\mathsf{pk}}))$ and an update token up, we explain how to update sk in UpdSk (Algorithm 7). The first step is to recover the randomness used to update the public key; to do so, we compute $\widetilde{\rho} \leftarrow \mathsf{Dec}(\mathsf{up}, \mathsf{sk})$. This $\widetilde{\rho}$ is then digested by H_{up}, which returns a scalar r determining the isogeny walk $\varphi_{\mathsf{up}} : E_{\mathsf{pk}} \to E'_{\mathsf{pk}}$ via $\mathsf{IsogenyWalk}$. Note that while the public key is required to be hashed by H_{up}, we do not give it as an explicit input to the algorithm as it can be recovered from the secret key.

Given the isogeny φ_{up}, we recover $\mathrm{End}(E'_{\mathsf{pk}})$ via $\mathsf{IsogenyToEnd}$. In order to run $\mathsf{ConstrainedIsogeny}$, we need an isogeny ψ of odd degree between E_0 and E'_{pk}. This can be computed as follows. If the previous secret isogeny had odd degree, then concatenating the isogeny representation of the secret isogeny with the update isogeny will suffice. Otherwise, we compute another odd degree isogeny from E_0 to E_{pk} (the ideal associated to $I_{\varphi_{\mathsf{sk}}}$ is known and from that one can compute an equivalent ideal of odd norm efficiently) and concatenate that to the update isogeny. This procedure also allows us to compute the ideal I_ψ associated to ψ.

Finally, we recover an isogeny representation $(\alpha'_{\mathrm{sk}}, \beta'_{\mathrm{sk}}, d'_{\mathsf{sk}}, s'_{\mathsf{sk}})$ of the updated secret isogeny $\varphi'_{\mathsf{sk}} : E_0 \to E'_{\mathsf{pk}}$ via $\mathsf{ConstrainedIsogeny}$. The updated secret key is given as $(\alpha'_{\mathrm{sk}}, \beta'_{\mathrm{sk}}, d'_{\mathsf{sk}}, s'_{\mathsf{sk}}, \mathrm{End}(E'_{\mathsf{pk}}))$.

5 Security

In this section, we will prove the security of our UPKE, which we refer to as Updatable FESTA. To this extent, we start by proving a few preliminary results. Namely, we show in Proposition 10 that FESTA-PKE, the PKE obtained using Algorithm 2 as KeyGen, Algorithm 3 as Enc and Algorithm 5 as Dec is OW-CPA. We then move to Updatable FESTA, proving its correctness in Proposition 12, γ-spreadness property in Theorem 13 and its OW-CR-CPA security in Theorem 15. Putting these results together, we apply a transform from [6, §4.3; 4.4] to get Theorem 16 that provides an IND-CU-CCA UPKE.

5.1 OW-CPA security of FESTA-PKE

To prove that FESTA-PKE is OW-CPA, we recall the CIST^2 problem from [8, Problem 8], tailoring it to our use case.

Algorithm 7 $\mathsf{UpdSk}(\mathsf{sk}, \mathsf{up})$

Require: Current secret key $\mathsf{sk} = (\alpha_{\mathsf{sk}}, \beta_{\mathsf{sk}}, d_{\mathsf{sk}}, s_{\mathsf{sk}}, \mathrm{End}(E_{\mathsf{pk}}))$, the updated public key $\mathsf{pk}' = E_{\mathsf{pk}'}$ and the update token up.
Ensure: Updated secret key $\mathsf{sk}' = (\alpha'_{\mathsf{sk}}, \beta'_{\mathsf{sk}}, d'_{\mathsf{sk}}, s'_{\mathsf{sk}}, \mathrm{End}(E_{\mathsf{pk}'}))$

Let $\varphi_{\mathsf{sk}} : E_0 \to E_{pk}$ be the isogeny corresponding to the isogeny representation given by the secret key
$\widetilde{\rho} \leftarrow \mathsf{Dec}(\mathsf{up}, \mathsf{sk})$.
$\widetilde{r} \leftarrow \mathsf{H}_{\mathsf{up}}(\widetilde{\rho}, E_{\mathsf{pk}})$.
$_, \varphi_{\mathsf{up}} \leftarrow \mathsf{IsogenyWalk}_3(E_{\mathsf{pk}}, \delta, \widetilde{r})$.
$\mathrm{End}(E_{\mathsf{pk}'}) \leftarrow \mathsf{IsogenyToEnd}(\mathrm{End}(E_{\mathsf{pk}}), \varphi_{\mathsf{up}})$
if $\deg(\varphi_{\mathsf{sk}})$ is even **then**
    $\varphi_{\mathsf{sk}} \leftarrow \varphi'_{\mathsf{sk}}$ where φ'_{sk} is a random isogeny from $E_0 \to E_{pk}$ of odd degree
$\psi \leftarrow \varphi_{\mathsf{up}} \circ \varphi_{\mathsf{sk}}$.
From the representation of $\mathrm{End}(E_{\mathsf{pk}'})$ get I_ψ.
$(\alpha'_{\mathsf{sk}}, \beta'_{\mathsf{sk}}, d'_{\mathsf{sk}}, s'_{\mathsf{sk}}) \leftarrow \mathsf{ConstrainedIsogeny}(E_{\mathsf{pk}'}, \mathrm{End}(E_{\mathsf{pk}'}), \psi, I_\psi)$.
$\mathsf{sk}' \leftarrow (\alpha'_{\mathsf{sk}}, \beta'_{\mathsf{sk}}, d'_{\mathsf{sk}}, s'_{\mathsf{sk}}, \mathrm{End}(E_{\mathsf{pk}'}))$.
return sk'.

Problem 9 (CIST^2*).* Let $E_0 : y^2 = x^3 + x$ be a supersingular elliptic curve defined over $\mathbb{F}_{p^2}$, and let E'_0 be a random supersingular elliptic curve defined over the same field. Consider two isogenies $\varphi : E_0 \to E_1$ and $\varphi' : E'_0 \to E'_1$ of smooth degrees 3^b and 5^c, respectively, and let $\alpha, \beta \in (\mathbb{Z}/N\mathbb{Z})^*$. Given the curves E_0, E_1, E'_0, E'_1 and the points

$$(P_1, Q_1) = ([\alpha]\varphi(P_0), [\beta]\varphi(Q_0)) \quad \text{and} \quad (P'_1, Q'_1) = ([\alpha]\varphi'(P'_0), [\beta]\varphi'(Q'_0)),$$

where $(P_0, Q_0) \leftarrow \mathsf{Gen2Tor}(E_0)$ and $(P'_0, Q'_0) \leftarrow \mathsf{Gen2Tor}(E'_0)$, compute some representations for the isogenies φ and φ'.

Proposition 10. *Under the* CIST^2 *assumption, FESTA-PKE is OW-CPA.*

Proof. Let $\mathcal{A}$ be an adversary against OW-CPA of FESTA-PKE, and consider an adversary $\mathcal{B}$ against CIST^2, with parameters $d = 3^b$, $d' = 5^c$ and $N = 2^a$.

On input a tuple $(E_0, E_1, E'_0, E'_1, P_1, Q_1, P'_1, Q'_1)$, the adversary $\mathcal{B}$ sets $\mathsf{pk} = E'_0$. We highlight that, thanks to the use of isogeny walks of length δ guaranteeing uniform distribution of public curves (Section 4.1), the curve E'_0 is indeed an admissible public key.

Then, the adversary $\mathcal{B}$ defines the challenge ciphertext to be

$$\mathsf{c}^* = (E_1, P_1, Q_1, E'_1, P'_1, Q'_1),$$

which has the correct distribution by direct construction, and calls $\mathcal{A}(\mathsf{pk}, \mathsf{c}^*)$. The adversary $\mathcal{B}$ finally collects the response m^*.

To recover φ, φ', $\mathcal{B}$ runs $\mathsf{MsgToScalars}(m^*) = (\alpha^*, \beta^*)$, which it can use to unmask $(P_1, Q_1), (P'_1, Q'_1)$ to get $\varphi(P_0), \varphi(Q_0)$, and $\varphi'(P'_0), \varphi'(Q'_0)$. This information gives HD representations for φ, φ', hence breaking CIST^2. □

Remark 11. Note that unlike FESTA [8] and QFESTA [43], we solely rely on the CIST^2 to prove the one-way security of FESTA-PKE and do not need the additional DIST assumption.

The DIST assumption was used to argue that public keys are indistinguishable from random elliptic curves. In our construction, public keys are truly uniform, hence strengthening our security claims.

It is also worth noting that given a public curve E_{pk}, the basis $(P_{\mathsf{pk}}, Q_{\mathsf{pk}})$ is not necessarily the image of $\langle P_0 \rangle$ and $\langle Q_0 \rangle$ under the isogeny generated by $\mathsf{IsogenyWalk}_3$. Concretely, the degree of the isogeny matching the constraint $(\alpha_{\mathrm{sk}}, \beta_{\mathrm{sk}})$ is secret. This removes important information that was shown to be crucial to define weak instances [23,17].

5.2 Correctness of Updatable FESTA

Proposition 12 (Correctness). *The UPKE described in Section 4 is correct.*

Proof. First, the correctness of FESTA-PKE follows directly from the correctness of FESTA [8, §3], the discussion in Sections 4.2 to 4.4, and our choice of parameters (Section 4.1).

Second, we need to show that the correctness of decryption is preserved by updates. This comes as a direct consequence of the correctness of the subroutines used in UpdSk (Algorithm 7). Given a valid key pair $(\mathsf{sk}_i, \mathsf{pk}_i)$, and an update randomness $\rho_i \leftarrow_{\$} \{0,1\}^{2\lambda}$, UpdPk sends $\mathsf{pk}_i = E_{\mathsf{pk}_i}$ to a new curve $\mathsf{pk}_{i+1} = E_{\mathsf{pk}_{i+1}}$ by performing a walk of length δ in the 3-isogeny graph described by $r_i \leftarrow \mathsf{H}_{\mathsf{up}}(\rho_i)$. By our choice of δ, the distribution of resulting curve will be close to uniform. Then given an honestly updated key $\mathsf{sk}_i = (\alpha_i, \beta_i, d_i, s_i, \mathrm{End}(E_{\mathsf{pk}_i}))$, and update token $\mathsf{up}_i = \mathsf{Enc}(\rho_i, \mathsf{pk}_i)$, one first computes $\mathsf{Dec}(\mathsf{up}_i, \mathsf{sk}_i)$. By the correctness of decryption for $(\mathsf{sk}_i, \mathsf{pk}_i)$ this is exactly ρ_i. Then one recomputes $E_{\mathsf{pk}_{i+1}}$ by performing the same isogeny walk φ_{up_i} described by $r_i \leftarrow \mathsf{H}_{\mathsf{up}}(\rho_i)$, and stores the corresponding isogeny, calls IsogenyToEnd on input $(\mathrm{End}(\mathsf{pk}_i), \varphi_{\mathsf{up}_i})$ to get $\mathrm{End}(E_{\mathsf{pk}_{i+1}})$ and finally calls ConstrainedIsogeny to obtain $(\alpha_{i+1}, \beta_{i+1}, d_{i+1}, s_{i+1})$. By the discussion of Section 3.1, ConstrainedIsogeny is correct which ensures that $\mathsf{sk}_{i+1} = (\alpha_{i+1}, \beta_{i+1}, d_{i+1}, s_{i+1}, \mathrm{End}(E_{\mathsf{pk}_{i+1}}))$ is well-formed. Thus $(\mathsf{sk}_{i+1}, \mathsf{pk}_{i+1})$ is a valid key pair, and by the correctness of decryption the statement is proven. □

5.3 OW-CR-CPA security and γ spreadness of Updatable FESTA

Theorem 13 (γ-spreadness). *The Updatable-FESTA UPKE scheme described in Section 4 satisfies the γ-spreadness property (Definition 21).*

Proof. Let $\mathsf{ctxt}^* = (E_1^*, R_1^*, S_1^*, E_2^*, R_2^*, S_2^*)$ be any fixed ciphertext, and let $\mathsf{ctxt} = (E_1, R_1, S_1, E_2, R_2, S_2)$ a random ciphertext returned by the encryption algorithm of our UPKE. Then

$$\Pr[\mathsf{ctxt} = \mathsf{ctxt}^*] \leq \Pr[E_1 = E_1^*, E_2 = E_2^*] = \Pr[E_1 = E_1^*] \cdot \Pr[E_2 = E_2^*],$$

where the last equality comes from the fact that E_1 and E_2 are independently generated in the encryption algorithm.

First, the curve E_1 is obtained by computing a 3^b-isogeny $\varphi_{\mathsf{Enc}_1} : E_0 \to E_1$ of kernel $\langle R_3 + [r_3]S_3 \rangle$ where r_3 is sampled uniformly from $[0, 3^b)$ and (R_3, S_3) is a fixed basis of $E_0[3^b]$. As in [44, Theorem 1], one can prove that $E_0 : y^2 = x^3 + x$ admits no endomorphism of degree 3^{2b} whenever $3^{2b} < \frac{p+1}{4}$, which is the case for our parameters. Hence, each choice of r_3 yields a distinct curve E_1, implying that

$$\Pr[E_1 = E_1^*] = \frac{1}{3^b} = 2^{-2\lambda}.$$

Second, $\varphi_{\mathsf{Enc}_2} : E_{\mathsf{pk}} \to E_2$ is a uniformly random isogeny of degree 5^c from E_{pk}, and there are $6 \cdot 5^{c-1} = 5^c + 5^{c-1} > 2^{2\lambda}$ such isogenies. From [39], the number of supersingular curves having an endomorphism of degree 5^{2c} is bounded by $O(5^{3c})$. Since E_{pk} is uniformly distributed in the supersingular isogeny graph by construction, then the probability that E_{pk} admits an endomorphism of degree 5^{2c} is bounded by $\frac{12 \cdot 5^{3c}}{p} \approx \frac{2^{6\lambda}}{2^{12\lambda}} \approx 2^{-6\lambda}$. We hence obtain

$$\begin{aligned}
\Pr[E_2 = E_2^*] &= \Pr[E_2 = E_2^*, \varphi_{\mathsf{Enc}_2} = \varphi^*_{\mathsf{Enc}_2}] + \Pr[E_2 = E_2^*, \varphi_{\mathsf{Enc}_2} \neq \varphi^*_{\mathsf{Enc}_2}] \\
&\leq \tfrac{1}{5^c + 5^{c-1}} + \Pr[\exists \theta \in \mathrm{End}(E_{\mathsf{pk}}), \deg(\theta) = 3^{2b}] \\
&\leq 2^{-2\lambda} + 2^{-6\lambda}.
\end{aligned}$$

Finally, we obtain

$$\begin{aligned}
\Pr[\mathsf{ctxt} = \mathsf{ctxt}^*] &\leq \Pr[E_1 = E_1^*] \cdot \Pr[E_2 = E_2^*] \\
&\leq 2^{-2\lambda} \cdot (2^{-2\lambda} + 2^{-6\lambda}) \\
&= 2^{-4\lambda} + 2^{-8\lambda},
\end{aligned}$$

which means that our UPKE is γ-spread with $\gamma \approx 4\lambda$. □

Proposition 14 (Key indistinguishability). *In Updatable FESTA, freshly generated key pairs, i.e., key pairs generated directly as* $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$ *are statistically indistinguishable from updated key pairs i.e., key pairs obtained as* $\big((\mathsf{pk}', \mathsf{up}), \mathsf{sk}'\big) \leftarrow (\mathsf{UpdPk}(\mathsf{pk}, \rho), \mathsf{UpdSk}(\mathsf{sk}, \mathsf{up}))$ *for some* $\rho \leftarrow_\$ \{0,1\}^{2\lambda}$.

Proof. First of all, in both cases the secret keys are obtained from of the public key following the same procedure which returns the shortest isogeny in $\mathrm{Hom}(E_0, E_{\mathsf{pk}})_{(P_0,Q_0),(P_{\mathsf{pk}},Q_{\mathsf{pk}})}$ where $(P, Q) \leftarrow \mathsf{Gen2Tor}(E_0)$ and $(P_{\mathsf{pk}}, Q_{\mathsf{pk}}) \leftarrow \mathsf{Gen2Tor}(E_{\mathsf{pk}})$, so we focus on the public key distribution. During key generation, the public key is generated as a random isogeny walk of length $\delta := \lceil \frac{15}{4} \log_3(p) \rceil$ in the 3-isogeny graph. As pointed out in Remark 7, by our choice of parameters this means that the public distribution is $O(2^{-16.5\lambda})$ close to the uniform distribution on the supersingular isogeny graph. By a similar argument, updated public keys are also generated as a walk of length δ in the 3-isogeny graph but starting from the previous public key. Nonetheless, the resulting distribution is equally close to the stationary one. Putting together these two results we get that the distributions of the fresh and updated public keys are $O(2 \cdot 2^{-16.5\lambda})$ close. □

When proving that our UPKE is OW-CR-CPA secure, we will at some point need to simulate an update that sends the public key E_{pk_i} of some epoch i to a freshly generated public key E_{pk}. This is achieved by generating an isogeny $\varphi : E_{\mathsf{pk}_i} \to E_{\mathsf{pk}}$ of degree 3^δ using KLPT, and translating this isogeny into a string r such that $E_{\mathsf{pk}} \leftarrow \mathsf{IsogenyWalk}(E_{\mathsf{pk}}, \delta, r)$. For the adversary to not be able to distinguish between such a fabricated update and a normal update, the outputs of the KLPT algorithm need to be indistinguishable from uniformly random strings $r = (r_0, r_1, \cdots, r_{\delta-1}) \in [0, \ell+1) \times [0, \ell)^{\delta-1}$, $\ell = 3$ in our case. This is the purpose of Assumption 1, which was also used in the zero-knowledge proof of the first version of SQIsign [1, Version 1.0, §9.1.3.2].

Assumption 1 ($\mathsf{IND\text{-}KLPT}$) *Let E be a uniformly random supersingular elliptic curve defined over $\mathbb{F}_{p^2}$ and let $\delta = \lceil \frac{15}{4} \log_\ell(p) \rceil$. For all PPT adversaries $\mathcal{A}$, the advantage $\mathsf{Adv}^{\mathsf{IND\text{-}KLPT}}(\mathcal{A})$ of $\mathcal{A}$ in distinguishing the following distributions is negligible:*

- *r, where $r = (r_0, r_1, \cdots, r_{\delta-1})$ is sampled uniformly at random from $[0, \ell+1) \times [0, \ell)^{\delta-1}$;*
- *r, where $r = (r_0, r_1, \cdots, r_{\delta-1}) \in [0, \ell+1) \times [0, \ell)^{\delta-1}$ is obtained as $r \leftarrow \mathsf{KLPT}_\ell(E, \mathrm{End}(E), E', \mathrm{End}(E'))$ with E' being a uniformly random supersingular elliptic curve.*

Theorem 15 (OW-CR-CPA security). *In the random oracle model, if Assumption 1 holds and the CIST^2 problem is hard, then the Updatable-FESTA UPKE scheme described in Section 4 is OW-CR-CPA (Definition 2).*

Proof. We preliminary remark that, under the assumption that the CIST^2 is hard, we have already proved that $\mathsf{FESTA\text{-}PKE}$ is OW-CPA in Proposition 10. Moreover, by Proposition 14, we have that fresh keys and updated keys are statistically indistinguishable.

Now, consider G_0 to be the original OW-CR-CPA game. We model H_{up} as a random oracle. So the adversary has access to both UpdO and RO. We also explicitly unroll the UpdPk procedure within the oracle.

We write out G_0 explicitly in Figure 3.

G_1: Our first goal is to reduce to a scenario where we can estimate the epoch q during which the adversary is going to make its challenge query for c^*. This is a standard guessing argument. We consider a first series of hybrid games H_i, where we keep the game essentially the same, except that the challenger in H_i makes a guess i in advance on the value of q for which the adversary is going to issue the first challenge query, and aborts and returns a random message if $\mathsf{q} \neq i$. When $i = \mathsf{q}$, the game is perfectly simulated. Otherwise, we output a random plaintext. Let q_1 be the maximal number of queries issued by $\mathcal{A}$ to UpdO in the first phase of the game (i.e., on line 3.). Then, by a standard hybrid argument $\mathsf{Adv}_{\mathsf{G}_0}(\mathcal{A}) \leq \sum_{i=0}^{q_1} \mathsf{Adv}_{H_i}(\mathcal{A})$. So in particular, there exists some i^* such that $\mathsf{Adv}_{\mathsf{G}_0}(\mathcal{A}) \leq (q_1+1)\mathsf{Adv}_{H_{i^*}}(\mathcal{A})$. Let $G_1 = H_{i^*}$. So at this stage, we assume the challenger has correctly guessed q.

$\mathsf{G}_0(\mathcal{A})$

$i \leftarrow 0$
$(\mathsf{pk}_0, \mathsf{sk}_0) \leftarrow \mathsf{KeyGen}(1^\lambda)$
$\mathsf{state} \leftarrow \mathcal{A}^{\mathsf{UpdO}(\rho),\mathsf{RO}(\rho)}(\mathsf{pk}_0)$
$\mathsf{q} \leftarrow i$
$\mathsf{mex}^* \leftarrow_\$ \mathcal{M}$
$\mathsf{ctxt}^* \leftarrow \mathsf{Enc}(\mathsf{pk}_\mathsf{q}, \mathsf{mex}^*)$
$\mathsf{state} \leftarrow \mathcal{A}^{\mathsf{UpdO}(\rho)}(\mathsf{ctxt}^*, \mathsf{state})$
$\mathsf{q}' \leftarrow i$
$\rho^* \leftarrow_\$ \{0,1\}^{2\lambda}$
$(\mathsf{pk}^*, \mathsf{up}^*) \leftarrow \mathsf{UpdPk}(\mathsf{pk}_{\mathsf{q}'}, \rho^*)$
$\mathsf{sk}^* \leftarrow \mathsf{UpdSk}(\mathsf{sk}_{\mathsf{q}'}, \mathsf{up}^*)$
$\mathsf{mex}' \leftarrow \mathcal{A}(\mathsf{ctxt}^*, \mathsf{pk}^*, \mathsf{sk}^*, \mathsf{up}^*, \mathsf{state})$
return $(\mathsf{mex}' = \mathsf{mex}^*)$

$\mathsf{UpdO}(\rho)$

$r \leftarrow \mathsf{RO}(\rho \,||\, \mathsf{E}_{\mathsf{pk}_i})$
$\mathsf{E}_{\mathsf{pk}_{i+1}}, _ \leftarrow \mathsf{IsogenyWalk}_3(\mathsf{E}_{\mathsf{pk}_i}, \delta, r)$
$\mathsf{up}_i \leftarrow \mathsf{Enc}(\mathsf{pk}_i, \rho)$
$\mathsf{sk}_{i+1} \leftarrow \mathsf{UpdSk}^{\mathsf{RO}}(\mathsf{sk}_i; \mathsf{up}_i)$
$i \leftarrow i + 1$

$\mathsf{RO}(\rho)$

if $\rho \notin \mathcal{Q}$: $\mathcal{Q}[\rho] \leftarrow_\$ [0, 3^\delta)$
return $\mathcal{Q}[\rho]$

Fig. 3: G_0

G_2: Now that we assume we have correctly guessed the value of q, we want to ensure that $(\mathsf{sk}_\mathsf{q}, \mathsf{pk}_\mathsf{q})$ is generated honestly, so that the challenge ciphertext comes from an honest key pair.

The idea is to let the adversary make their first $\mathsf{q} - 1$ queries to UpdO freely, and then use the programmability of the random oracle to force the last update of the adversary to end up on a target, honestly generated public key $(\mathsf{sk}_\mathsf{q}, \mathsf{pk}_\mathsf{q}) \leftarrow \mathsf{KeyGen}(1^\lambda)$. This is possible because the challenger has knowledge of the entire secret key chain $\mathsf{sk}_0 \rightarrow ... \rightarrow \mathsf{sk}_{\mathsf{q}-1}$ as well as sk_q. The secret keys contain the knowledge of the endomorphism ring of the corresponding curves, therefore, the challenger can use the KLPT algorithm, as described in Section 2.2, to compute an isogeny between $E_{\mathsf{pk}_{\mathsf{q}-1}}$ and $E_{\mathsf{pk}_\mathsf{q}}$ of degree 3^δ [9]. Then, they encode that isogeny as the corresponding CGL path value r_q which is given as input to $\mathsf{IsogenyWalk}$, i.e., we have $E_{\mathsf{pk}_\mathsf{q}} \leftarrow \mathsf{IsogenyWalk}(E_{\mathsf{pk}_{\mathsf{q}-1}}, \delta, r_\mathsf{q})$. Having knowledge of this r_q value, all that is left to do is to embed it as the answer of the qth query of $\mathcal{A}$ to UpdO, i.e., we program the random oracle such that $H(\rho_\mathsf{q} \,||\, E_{\mathsf{pk}_{\mathsf{q}-1}}) = r_\mathsf{q}$, where ρ_q denotes the value given as input to UpdO at the time of the adversary q^{th} query. The probability that the adversary notices that the RO is being reprogrammed is bounded by $\mathsf{Adv}^{\mathsf{IND\text{-}KLPT}}(\mathcal{A})$. However, the adversary also has access to the random oracle, and could have previously queried this same value, which prevents us from programming it freely. To address this, we use a standard query-guessing argument. Let q_H denote the number of queries made by $\mathcal{A}$ to RO in this epoch. Then we need to make a guess on the

[9] This step justifies our choice of a large value for δ.

index i^* on which of the q^H possible RO queries will also be made to UpdO and plant our value r_{q} at this i^* query. If we make a wrong guess or if the query to UpdO is actually fresh, we abort. The probability of correctly guessing i^* is therefore $\frac{1}{q_H+1}$, which leads to a security loss factor of (q_H+1). We hence have $\mathsf{Adv}(G_1) \leq (q_H+1)\left(\mathsf{Adv}(G_2) + \mathsf{Adv}^{\mathsf{IND\text{-}KLPT}}(\mathcal{A})\right)$.

So now at the end we arrive at an equivalent game G_2 where $(\mathsf{sk}_{\mathsf{q}}, \mathsf{pk}_{\mathsf{q}}) \leftarrow \mathsf{KeyGen}$ directly, so we can ignore the first phase. We summarize it in Figure 4.

$\mathsf{G}_2(\mathcal{A})$

1 : $(\mathsf{sk}_{\mathsf{q}}, \mathsf{pk}_{\mathsf{q}}) \leftarrow \mathsf{KeyGen}(1^\lambda)$
2 : $\mathsf{mex}^* \leftarrow_{\$} \mathcal{M}$
3 : $\mathsf{ctxt}^* \leftarrow \mathsf{Enc}(\mathsf{pk}_{\mathsf{q}}, \mathsf{mex}^*)$
4 : $\mathsf{state} \leftarrow \mathcal{A}^{\mathsf{UpdO}(\rho)}(\mathsf{pk}_{\mathsf{q}}, \mathsf{ctxt}^*)$
5 : $\mathsf{q}' \leftarrow i$
6 : $\rho^* \leftarrow_{\$} \{0,1\}^{2\lambda}$
7 : $(\mathsf{pk}^*, \mathsf{up}^*) \leftarrow \mathsf{UpdPk}(\mathsf{pk}_{\mathsf{q}'}, \rho^*)$
8 : $\mathsf{sk}^* \leftarrow \mathsf{UpdSk}(\mathsf{sk}_{\mathsf{q}'}, \mathsf{up}^*)$
9 : $\mathsf{mex}' \leftarrow \mathcal{A}(\mathsf{ctxt}^*, \mathsf{pk}^*, \mathsf{sk}^*, \mathsf{up}^*, \mathsf{state})$
10 : **return** $(\mathsf{mex}' = \mathsf{mex}^*)$

$\mathsf{UpdO}(\rho)$

1 : $r \leftarrow \mathsf{RO}(\rho \,||\, E_{\mathsf{pk}_i})$
2 : $\mathsf{E}_{\mathsf{pk}_{i+1}}, _ \leftarrow \mathsf{IsogenyWalk}_3(E_{\mathsf{pk}_i}, \delta, r)$
3 : $\mathsf{up}_i \leftarrow \mathsf{Enc}(\mathsf{pk}_i, \rho)$
4 : $\mathsf{sk}_{i+1} \leftarrow \mathsf{UpdSk}^{\mathsf{RO}}(\mathsf{sk}_i, \mathsf{up}_i)$
5 : $i \leftarrow i+1$

$\mathsf{RO}(\rho)$

1 : **if** $\rho \notin \mathcal{Q}$: $\mathcal{Q}[\rho] \leftarrow_{\$} [0, 3^\delta)$
2 : **return** $\mathcal{Q}[\rho]$

Fig. 4: G_2

G_3: This is the same as G_2 except that we compute $(\mathsf{sk}^*, \mathsf{pk}^*)$ as an update from $(\mathsf{sk}_{\mathsf{q}}, \mathsf{pk}_{\mathsf{q}})$ directly, rather than from $(\mathsf{sk}_{\mathsf{q}'}, \mathsf{pk}_{\mathsf{q}'})$ using some randomness ρ^* which is sampled at the beginning of the game. The difference in the two games is bounded by the statistical distance between the distributions of the two curves which, by a similar reasoning as that of Section 5.3 is bounded by $2^{-16.5\lambda+1}$. Hence, we get $|\mathsf{Adv}_{\mathsf{G}_3} - \mathsf{Adv}_{\mathsf{G}_2}| \leq 2^{-16.5\lambda+1}$. Furthermore, note that since the leaked key pair is computed directly from $(\mathsf{sk}_{\mathsf{q}}, \mathsf{pk}_{\mathsf{q}})$ the updates are no longer used in the game.

G_4: This is the same as G_3 except that we remove the adversary's access to UpdO and instead we can perfectly simulate it using RO, and manually updating the public key. The advantage remains the same.

G_5: This is the same as G_4 except that if the adversary queries $\mathsf{RO}(\rho^*)$ (recall ρ^* is used to update pk_{q} to pk^*), the game returns 1. We have $\mathsf{Adv}(G_5) \leq \mathsf{Adv}(G_4) + \frac{1}{2^{2\lambda}}$, accounting for the probability of the adversary guessing $\rho^* \in \{0,1\}^{2\lambda}$.

G_6: This is the same as G_5 except that instead of having $r \leftarrow \mathsf{RO}(\rho^*)$, we directly compute $r \leftarrow_\$ [0, 3^\delta)$. Since in the random oracle model r is independent of ρ^* until ρ^* is queried, the advantage stays the same. (Indeed, we took care of the case where ρ^* is queried in the previous game).

G_7: So now, we are ready to replace $(\mathsf{sk}^*, \mathsf{pk}^*)$ by a freshly generated key pair instead of an update from $(\mathsf{sk}_\mathsf{q}, \mathsf{pk}_\mathsf{q})$. The difference in advantage between these two games is bounded by the statistical distance between the distributions of freshly generated keys and updated keys. By Section 5.3, and our parameter choice, this distance is negligible. Hence, we have

$$|\mathsf{Adv}(\mathsf{G}_7) - \mathsf{Adv}(\mathsf{G}_6)| \leq 2 \cdot 2^{-16.5\lambda}.$$

Let us summarize the game in Figure 5

$\mathsf{G}_7(\mathcal{A})$	$\mathsf{RO}(\rho)$
1 : $\rho^* \leftarrow_\$ \{0,1\}^{2\lambda}$	1 : **if** $\rho = \rho^*$: **FreeWin** $= 1$ **return** $\perp$
2 : $(\mathsf{sk}_\mathsf{q}, \mathsf{pk}_\mathsf{q}) \leftarrow \mathsf{KeyGen}(1^\lambda)$	2 : **if** $\rho \notin \mathcal{Q}$: $\mathcal{Q}[\rho] \leftarrow_\$ [0, 3^\delta)$
3 : $\mathsf{mex}^* \leftarrow_\$ \mathcal{M}$	3 : **return** $\mathcal{Q}[\rho]$
4 : $\mathsf{ctxt}^* \leftarrow \mathsf{Enc}(\mathsf{pk}_\mathsf{q}, \mathsf{mex}^*)$	
5 : $\mathsf{state} \leftarrow \mathcal{A}^{\mathsf{RO}}(\mathsf{pk}_\mathsf{q}, \mathsf{ctxt}^*)$	
6 : $(\mathsf{pk}^*, \mathsf{sk}^*) \leftarrow \mathsf{KeyGen}(1^\lambda)$	
7 : $\mathsf{up}^* \leftarrow \mathsf{Enc}(\mathsf{pk}_\mathsf{q}, \rho^*)$	
8 : $\mathsf{mex}' \leftarrow \mathcal{A}(\mathsf{ctxt}^*, \mathsf{pk}^*, \mathsf{sk}^*, \mathsf{up}^*, \mathsf{state})$	
9 : **return** $(\mathsf{mex}' = \mathsf{mex}^*)$ or **FreeWin**	

Fig. 5: G_7

Now we argue that G_7 is essentially the same as the OW-CPA game for FESTA. The challenge ciphertext is computed from an honest and freshly generated key pair, as in the OW-CPA game for FESTA-PKE. The random oracle can be simulated with lazy sampling and the exposed key pair also comes from a freshly generated key generation, which can be perfectly simulated. Finally, up^* is also the encryption of a random value which can be simulated. So, for any adversary $\mathcal{B}$ against OW-CPA game for FESTA-PKE, we have $\mathsf{Adv}_{\mathsf{G}_7}(\mathcal{A}) \leq \mathsf{Adv}^{\mathsf{OW\text{-}CPA}}_{\mathsf{FESTA\text{-}PKE}}(\mathcal{B})$.

Thus, by unrolling this series of games, given adversaries $\mathcal{A}$ and $\mathcal{B}$ against OW-CR-CPA of the UPKE and OW-CPA of FESTA-PKE respectively, we have

$$\mathsf{Adv}^{\mathsf{OW\text{-}CR\text{-}CPA}}_{\mathsf{FESTA\text{-}UPKE}}(\mathcal{A}) \leq (q_1+1)(q_H+1)\left(\mathsf{Adv}^{\mathsf{OW\text{-}CPA}}_{\mathsf{FESTA\text{-}PKE}}(\mathcal{B}) + \mathsf{Adv}^{\mathsf{IND\text{-}KLPT}}(\mathcal{A}) + \frac{2}{2^{16.5\lambda+1}} + \frac{1}{2^{2\lambda}}\right).$$

By assumption, $\mathsf{Adv}^{\mathsf{OW\text{-}CPA}}_{\mathsf{FESTA\text{-}PKE}}(\mathcal{B})$ and $\mathsf{Adv}^{\mathsf{IND\text{-}KLPT}}(\mathcal{A})$ are negligible and q_1, q_H are polynomial in the security parameter, which proves the statement. □

On the IND-KLPT *assumption.* De Feo et al. [25] analyzed the variant of KLPT used in the original design of SQIsign [24]. They showed that the IND-KLPT assumption did not hold for that variant of KLPT when $\ell = 2$ and proposed a new variant which they further analyzed [25, Section 6.2] and provided experimental evidence that their proposed fix prevents distinguishing attacks. While we use $\ell = 3$ (meaning that the distinguisher described in [25] would not apply), this analysis increases our confidence regarding the IND-KLPT assumption. More recently, Borin, Lai and Leroux [15] analyzed KLPT and showed that when the adversary has access to the secret key, they can distinguish KLPT outputs from random isogenies of the same degree. They then proposed a new variant of KLPT in which the distinguishability issue is addressed. In this new variant, the length of the output of KLPT is $\delta \approx \frac{9}{2}\log(p) + \frac{\lambda}{2} + O(\log\log(p))$. One should note that in our setting, the adversary does not have access to the secret key. Nevertheless, the analysis in [15] suggests that in a setting where the adversary does have access to the secret key, increasing δ from $\lceil \frac{15}{4}\log(p) \rceil$ to $\frac{9}{2}\log(p) + \frac{\lambda}{2} + O(\log\log(p))$ would be sufficient.

5.4 IND-CU-CCA security

We construct an IND-CU-CCA-secure Festa-based UPKE scheme $\Sigma := (\mathsf{KeyGen}', \mathsf{Enc}', \mathsf{Dec}', \mathsf{UpdPk}', \mathsf{UpdSk}')$ detailed in Figure 6 based on our OW-CR-CPA-secure UPKE scheme $\Pi := (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{UpdPk}, \mathsf{UpdSk})$ and an OT-CPA-secure symmetric scheme $\Gamma := (\mathsf{SEnc}, \mathsf{SDec})$ together with four random oracles described as follows:

- $G : \{0,1\}^* \to \mathcal{SK}_{\mathrm{sym}}$,
- $H : \{0,1\}^* \times \{0,1\}^* \times \{0,1\}^* \to \mathcal{R} = \{0,1\}^{2\lambda}$,
- $\widehat{G} : \{0,1\}^* \to \mathcal{SK}_{\mathrm{sym}}$,
- $\widehat{H} : \{0,1\}^* \times \{0,1\}^* \to \mathcal{R}$.

where $\mathcal{R}$, and $\mathcal{SK}_{\mathrm{sym}}$ are the spaces of randomness of Π, and the secret keys of Γ, respectively.

Theorem 16. *Let Π be the UPKE scheme described in Section 4 and let $\Gamma := (\mathsf{SEnc}, \mathsf{SDec})$ be a symmetric key encryption scheme that satisfies OT-CPA security. Then the UPKE scheme Σ obtained by applying the transformation of [6, §4.2] satisfies IND-CU-CCA security (Definition 22).*

Proof. The result follows from putting together Theorem 13 and Theorem 15, and Proposition 23. The proof is detailed in [6, §4.3; 4.4]. □

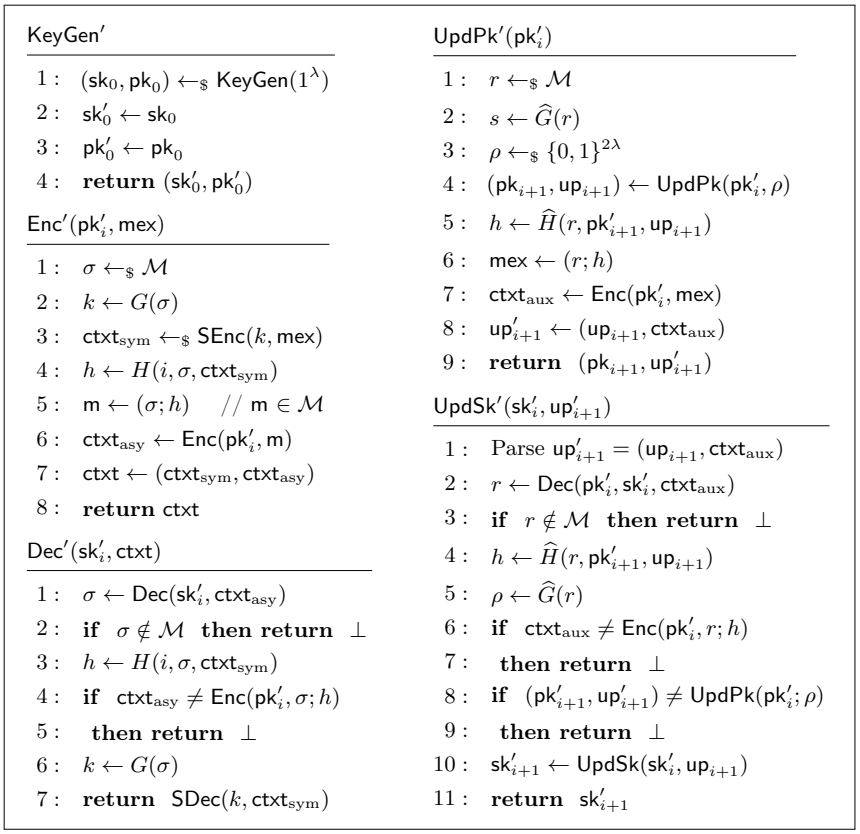

Fig. 6: IND-CU-CCA Festa-UPKE Scheme Σ.

On post-compromise security. Post-compromise security ensures that in a scenario where a secret key is exposed at a certain epoch e, all messages exchanged at epoch $e+1$ and beyond are secure. This captures the ability of the system to heal following key updates that occur after a key compromise. Our construction, like most UPKEs, does not naturally achieve post-compromise security. In fact, if the update token up is sent over a public channel, then an adversary having access to sk_i can retrieve up and run the secret key update algorithm to obtain sk_{i+1}. Nevertheless, in our construction, updated public keys and freshly generated ones have the same distribution. This implies that a fresh update from the legitimate owner of the secret key sk_i heals the system, as long as they can distribute the updated public key to the other parties through a separate authenticated channel. Moreover, an honest third party could also initiate an update and send the update token to the legitimate owner of the secret key sk_i through a separate secure (encrypted and authenticated) channel. The legitimate owner of the secret key sk_i then updates their key material to $(\mathsf{sk}_{i+1}, \mathsf{pk}_{i+1})$ and distributes pk_{i+1} to the other parties through an authenticated channel. Some variants of UPKE,

such as symmetric UPKEs [29], directly assume the existence of a secure channel that is used to send the update token to the receiver. We refer to [36, Section 4] for more details about achieving post-compromise security.

On forking security. Another security notion introduced by [4] for UPKE constructions is *forking security* (following the terminology of [41]). It is specifically tailored to the group messaging setting, in which not all users might have the same view, and updates to a key might be performed when someone is offline resulting in different views. Therefore, the series of updates would take the shape of an update tree rather than an update list as is currently the case in all security notions described so far. This security notion with an update tree is however only defined in the IND-CCA model, and is referred to more formally in [41] as Tree-Based IND-CCA. In our case, we build our protocol and prove its security in the OW-CR-CPA model, and lift its security to IND-CU-CCA through a generic transform, as is done for most post-quantum UPKE proposals. The notion of forking security is not defined in the one-way model, and it is beyond the scope of this work to do so. Similarly, there is no known transform between IND-CU-CCA and this notion (See [41], Figure 4). Furthermore, the notion of forking security is heavily tailored to the group messaging setting. In fact, there is no known post-quantum construction that achieves it, as [5] only considers a relaxation of this notion. Therefore, we leave the question as to whether or not our scheme could achieve such security open.

5.5 Results

We now instantiate our UPKE by providing actual parameters for the NIST security level I and discussing the key and ciphertext sizes. We use the 1631-bit prime $p_{1631} = f \cdot 2^a \cdot 3^b \cdot 5 - 1$ where

$$a = 1367,\ b = 162,\ f = 21.$$

The degree of the second encryption isogeny φ_{Enc_2} is 5^c where $c := 111$, and the degree of the update isogeny is 3^δ where $\delta = 3858$.

The public key is a supersingular elliptic curve E_{pk} defined over $\mathbb{F}_{p^2}$. It can be represented by its Montgomery coefficient using $2\log_2(p)$ bits. The ciphertext is a tuple $(E_1, R_1, S_1, E_2, R_2, S_2)$ where E_1 and E_2 are supersingular elliptic curves defined over $\mathbb{F}_{p^2}$, R_i and S_i are points of order 2^a on E_i for $i \in \{1, 2\}$. As before, the curves E_i can be represented by their Montgomery coefficient using $2\log_2(p)$ bits. The points R_i, S_i can be represented using their coordinates in the deterministic basis $(P_i, Q_i) \leftarrow \mathsf{Gen2Tor}(E_i)$ of $E_i[2^a]$, at the cost of some discrete logarithm computations. Each point is hence represented by two a bits long scalars. The total size of the ciphertext is hence $4\log_2(p) + 8a$ bits. For our NIST level I prime p_{1631}, the public key is 408 bytes while the ciphertext and the update token are 2183 bytes each. We compare the public key, ciphertext and update token sizes of isogeny-based constructions in Table 3.

UPKE	$\log p$	$\lvert\mathsf{pk}\rvert$	$\lvert\mathsf{ct}\rvert$	$\lvert\mathsf{up}\rvert$
Group action [29,35]	2,031	0.25	0.3	0.31
Group action [29,35]	4,089	0.5	0.55	0.55
SILBE [28]	13,013	3.18	7.95	7.95
This paper	1,631	0.4	2.13	2.13

Table 3: Communication cost (KiB) of isogeny-based UPKE schemes at the 128-bit security level. Here we assume that the CSIDH constructions, which are highly similar to each other in terms of sizes and performance, are instantiated with the 2,031 and 4,089 bits qt-PEGASIS [22] primes.

Next, we provide arguments to support the fact that our UPKE is several orders of magnitude more efficient compared to SILBE, the only isogeny-based UPKE which does not rely on isogeny group actions. In fact, in SILBE, the HD isogeny representations use dimension 4 isogenies of degree $N = 5\times7\times\cdots\times6863 \sim 2^{9765}$. As discussed in [28, §5.2], the decryption and the secret key update mechanism in SILBE are expected to cost at least $7^5\lambda^5\log(\lambda)^4$ and $2^33^5\lambda^5\log(\lambda)^4$ field operations over $\mathbb{F}_{p^2}$ where $p \sim 2^{13013}$. For $\lambda = 128 = 2^7$, we get about 2^{60} and 2^{57} field operations over $\mathbb{F}_{p^2}$ respectively, meaning that encryption and key updates will take several years to complete. In contrast, in our UPKE, the HD representations use dimension 4 isogenies of degree a multiple of 2^a with $a = 1367$ and the computations are done over a field of characteristic $p \sim 2^{1631}$. A partial implementation of our scheme, containing Encryption and ConstrainedIsogeny (Algorithm 1) is available at https://github.com/LucianoMaino/FESTA-UPKE.

6 Conclusions

In this work, we introduce a novel updatable public-key encryption scheme based on a 4-dimensional version of FESTA. One of its main advantages is that updated public keys are provably uniformly distributed among all supersingular elliptic curves. This feature allows us to prove its security relying on the hardness of the CIST^2 problem which also underlies the security of FESTA [8] and on the IND-KLPT assumption (Assumption 1) which states that outputs of the KLPT algorithm are indistinguishable from random isogenies of the same degree.

Our construction supports unbounded updates and does not rely on class group actions. We describe parameters that match NIST level security. We provide arguments supporting the fact that our UPKE is several orders of magnitude more efficient compared to its predecessor SILBE, the only prior isogeny-based UPKE which is not group action-based. While an implementation of Encryption and ConstrainedIsogeny (Algorithm 1) is provided, we leave a full implementation of our scheme for future work.

References

1. Aardal, M.A., Adj, G., Aranha, D.F., Basso, A., Canales Martínez, I.A., Chávez-Saab, J., Santos, M.C., Dartois, P., De Feo, L., Duparc, M., Eriksen, J.K., Fouotsa, T.B., Filho, D.L.G., Hess, B., Kohel, D., Leroux, A., Longa, P., Maino, L., Meyer, M., Nakagawa, K., Onuki, H., Panny, L., Patranabis, S., Petit, C., Pope, G., Reijnders, K., Robert, D., Rodríguez Henríquez, F., Schaeffler, S., Wesolowski, B.: SQIsign. Tech. rep., National Institute of Standards and Technology (2024), available at https://csrc.nist.gov/Projects/pqc-dig-sig/round-2-additional-signatures
2. Albrecht, M.R., Benčina, B., Lai, R.W.F.: Hollow LWE: A new spin - unbounded updatable encryption from LWE and PCE. In: Fehr, S., Fouque, P.A. (eds.) Advances in Cryptology – EUROCRYPT 2025, Part VIII. Lecture Notes in Computer Science, vol. 15608, pp. 363–392. Springer, Cham, Switzerland, Madrid, Spain (May 4–8, 2025). https://doi.org/10.1007/978-3-031-91101-9_13
3. Alwen, J., Coretti, S., Dodis, Y., Tselekounis, Y.: Security analysis and improvements for the IETF MLS standard for group messaging. In: Micciancio, D., Ristenpart, T. (eds.) Advances in Cryptology – CRYPTO 2020, Part I. Lecture Notes in Computer Science, vol. 12170, pp. 248–277. Springer, Cham, Switzerland, Santa Barbara, CA, USA (Aug 17–21, 2020). https://doi.org/10.1007/978-3-030-56784-2_9
4. Alwen, J., Fuchsbauer, G., Mularczyk, M.: Updatable public-key encryption, revisited. In: Joye, M., Leander, G. (eds.) Advances in Cryptology – EUROCRYPT 2024, Part VII. Lecture Notes in Computer Science, vol. 14657, pp. 346–376. Springer, Cham, Switzerland, Zurich, Switzerland (May 26–30, 2024). https://doi.org/10.1007/978-3-031-58754-2_13
5. Alwen, J., Fuchsbauer, G., Mularczyk, M., Riepel, D.: Lattice-based updatable public-key encryption for group messaging. Cryptology ePrint Archive, Paper 2025/365 (2025), https://eprint.iacr.org/2025/365
6. Asano, K., Watanabe, Y.: Updatable public key encryption with strong cca security: Security analysis and efficient generic construction. In: Patra, A. (ed.) Topics in Cryptology – CT-RSA 2025. pp. 223–246. Springer Nature Switzerland, Cham (2025)
7. Basso, A., Codogni, G., Connolly, D., De Feo, L., Fouotsa, T.B., Lido, G.M., Morrison, T., Panny, L., Patranabis, S., Wesolowski, B.: Supersingular curves you can trust. In: Hazay, C., Stam, M. (eds.) Advances in Cryptology – EUROCRYPT 2023, Part II. Lecture Notes in Computer Science, vol. 14005, pp. 405–437. Springer, Cham, Switzerland, Lyon, France (Apr 23–27, 2023). https://doi.org/10.1007/978-3-031-30617-4_14
8. Basso, A., Maino, L., Pope, G.: FESTA: Fast encryption from supersingular torsion attacks. In: Guo, J., Steinfeld, R. (eds.) Advances in Cryptology – ASIACRYPT 2023, Part VII. Lecture Notes in Computer Science, vol. 14444, pp. 98–126. Springer, Singapore, Singapore, Guangzhou, China (Dec 4–8, 2023). https://doi.org/10.1007/978-981-99-8739-9_4
9. Battagliola, M., Mora, R., Santini, P.: Using the Schur Product to Solve the Code Equivalence Problem. IEEE Transactions on Information Theory **72**(7), 4675–4694 (2026). https://doi.org/10.1109/TIT.2026.3694630
10. Bernstein, D.J., De Feo, L., Leroux, A., Smith, B.: Faster computation of isogenies of large prime degree. Open Book Series **4**(1), 39–55 (2020)
11. Bernstein, D.J., Lange, T., Martindale, C., Panny, L.: Quantum circuits for the CSIDH: Optimizing quantum evaluation of isogenies. In: Ishai, Y., Rijmen, V. (eds.) Advances in Cryptology – EUROCRYPT 2019, Part II. Lecture Notes in Computer

Science, vol. 11477, pp. 409–441. Springer, Cham, Switzerland, Darmstadt, Germany (May 19–23, 2019). https://doi.org/10.1007/978-3-030-17656-3_15
12. Boneh, D., Lewi, K., Montgomery, H.W., Raghunathan, A.: Key homomorphic prfs and their applications. In: Canetti, R., Garay, J.A. (eds.) Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part I. pp. 410–428. Lecture Notes in Computer Science, Springer (2013). https://doi.org/10.1007/978-3-642-40041-4_23, https://doi.org/10.1007/978-3-642-40041-4_23
13. Bonnetain, X., Schrottenloher, A.: Quantum security analysis of CSIDH. In: Canteaut, A., Ishai, Y. (eds.) Advances in Cryptology – EUROCRYPT 2020, Part II. Lecture Notes in Computer Science, vol. 12106, pp. 493–522. Springer, Cham, Switzerland, Zagreb, Croatia (May 10–14, 2020). https://doi.org/10.1007/978-3-030-45724-2_17
14. Borin, G., De Feo, L., Lido, G., Schaeffler, S.: The SQInstructor: a Guide to SQIsign and the Deuring Correspondence with Level Structures. In: Heninger, N., Rosulek, M. (eds.) Advances in Cryptology – CRYPTO 2026. pp. 537–569. Springer Nature Switzerland, Cham (2026). https://doi.org/10.1007/978-3-032-35398-6_18
15. Borin, G., Lai, Y.F., Leroux, A.: Erebor and durian: Full anonymous ring signatures from quaternions and isogenies. IACR Communications in Cryptology **1**(4) (2025). https://doi.org/10.62056/ava3zivrzn
16. Castryck, W., Decru, T.: An efficient key recovery attack on SIDH. In: Hazay, C., Stam, M. (eds.) Advances in Cryptology – EUROCRYPT 2023, Part V. Lecture Notes in Computer Science, vol. 14008, pp. 423–447. Springer, Cham, Switzerland, Lyon, France (Apr 23–27, 2023). https://doi.org/10.1007/978-3-031-30589-4_15
17. Castryck, W., Vercauteren, F.: A polynomial time attack on instances of M-SIDH and FESTA. In: Guo, J., Steinfeld, R. (eds.) Advances in Cryptology – ASIACRYPT 2023, Part VII. Lecture Notes in Computer Science, vol. 14444, pp. 127–156. Springer, Singapore, Singapore, Guangzhou, China (Dec 4–8, 2023). https://doi.org/10.1007/978-981-99-8739-9_5
18. Charles, D.X., Lauter, K.E., Goren, E.Z.: Cryptographic hash functions from expander graphs. Journal of Cryptology **22**, 93–113 (2009). https://doi.org/10.1007/s00145-007-9002-x
19. Chávez-Saab, J., Chi-Domínguez, J.J., Jaques, S., Rodríguez-Henríquez, F.: The SQALE of CSIDH: sublinear Vélu quantum-resistant isogeny action with low exponents. Journal of Cryptographic Engineering **12**(3), 349–368 (2022), https://doi.org/10.1007/s13389-021-00271-w
20. Corte-Real Santos, M., Eriksen, J.K., Meyer, M., Reijnders, K.: AprèsSQI: Extra fast verification for SQIsign using extension-field signing. In: Joye, M., Leander, G. (eds.) Advances in Cryptology – EUROCRYPT 2024, Part I. Lecture Notes in Computer Science, vol. 14651, pp. 63–93. Springer, Cham, Switzerland, Zurich, Switzerland (May 26–30, 2024). https://doi.org/10.1007/978-3-031-58716-0_3
21. Dartois, P., Eriksen, J.K., Fouotsa, T.B., Merdy, A.H.L., Invernizzi, R., Robert, D., Rueger, R., Vercauteren, F., Wesolowski, B.: PEGASIS: Practical effective class group action using 4-dimensional isogenies. In: Kalai, Y.T., Kamara, S.F. (eds.) Advances in Cryptology – CRYPTO 2025, Part I. Lecture Notes in Computer Science, vol. 16000, pp. 67–99. Springer, Cham, Switzerland, Santa Barbara, CA, USA (Aug 17–21, 2025). https://doi.org/10.1007/978-3-032-01855-7_3
22. Dartois, P., Eriksen, J.K., Invernizzi, R., Vercauteren, F.: qt-pegasis: Simpler and faster effective class group actions. In: Daemen, J., Thomé, E. (eds.) Advances in Cryptology – EUROCRYPT 2026. pp. 636–665. Springer Nature Switzerland, Cham (2026)

23. De Feo, L., Fouotsa, T.B., Panny, L.: Isogeny problems with level structure. In: Joye, M., Leander, G. (eds.) Advances in Cryptology – EUROCRYPT 2024, Part VII. Lecture Notes in Computer Science, vol. 14657, pp. 181–204. Springer, Cham, Switzerland, Zurich, Switzerland (May 26–30, 2024). https://doi.org/10.1007/978-3-031-58754-2_7
24. De Feo, L., Kohel, D., Leroux, A., Petit, C., Wesolowski, B.: SQISign: Compact post-quantum signatures from quaternions and isogenies. In: Moriai, S., Wang, H. (eds.) Advances in Cryptology – ASIACRYPT 2020, Part I. Lecture Notes in Computer Science, vol. 12491, pp. 64–93. Springer, Cham, Switzerland, Daejeon, South Korea (Dec 7–11, 2020). https://doi.org/10.1007/978-3-030-64837-4_3
25. De Feo, L., Leroux, A., Longa, P., Wesolowski, B.: New algorithms for the Deuring correspondence - towards practical and secure SQISign signatures. In: Hazay, C., Stam, M. (eds.) Advances in Cryptology – EUROCRYPT 2023, Part V. Lecture Notes in Computer Science, vol. 14008, pp. 659–690. Springer, Cham, Switzerland, Lyon, France (Apr 23–27, 2023). https://doi.org/10.1007/978-3-031-30589-4_23
26. Deuring, M.: Die typen der multiplikatorenringe elliptischer funktionenkörper: G. herglotz zum 60. geburtstag gewidmet. In: Abhandlungen aus dem mathematischen Seminar der Universität Hamburg. vol. 14, pp. 197–272. Springer (1941)
27. Dodis, Y., Karthikeyan, H., Wichs, D.: Updatable public key encryption in the standard model. In: Nissim, K., Waters, B. (eds.) TCC 2021: 19th Theory of Cryptography Conference, Part III. Lecture Notes in Computer Science, vol. 13044, pp. 254–285. Springer, Cham, Switzerland, Raleigh, NC, USA (Nov 8–11, 2021). https://doi.org/10.1007/978-3-030-90456-2_9
28. Duparc, M., Fouotsa, T.B., Vaudenay, S.: SILBE: An updatable public key encryption scheme from lollipop attacks. In: Eichlseder, M., Gambs, S. (eds.) SAC 2024: 31st Annual International Workshop on Selected Areas in Cryptography, Part I. Lecture Notes in Computer Science, vol. 15516, pp. 151–177. Springer, Cham, Switzerland, Montreal, Canada (Aug 28-30, 2024). https://doi.org/10.1007/978-3-031-82852-2_7
29. Eaton, E., Jao, D., Komlo, C., Mokrani, Y.: Towards post-quantum key-updatable public-key encryption via supersingular isogenies. In: AlTawy, R., Hülsing, A. (eds.) SAC 2021: 28th Annual International Workshop on Selected Areas in Cryptography. Lecture Notes in Computer Science, vol. 13203, pp. 461–482. Springer, Cham, Switzerland, Virtual Event (Sep 29 – Oct 1, 2022). https://doi.org/10.1007/978-3-030-99277-4_22
30. Eriksen, J.K., Panny, L., Sotáková, J., Veroni, M.: Deuring for the people: supersingular elliptic curves with prescribed endomorphism ring in general characteristic. In: LuCaNT: LMFDB, Computation, and Number Theory, Contemporary Mathematics, vol. 796, pp. 339–373. American Mathematical Society (2023)
31. Fouotsa, T.B., Kutas, P., Merz, S.P., Ti, Y.B.: On the isogeny problem with torsion point information. In: Hanaoka, G., Shikata, J., Watanabe, Y. (eds.) PKC 2022: 25th International Conference on Theory and Practice of Public Key Cryptography, Part I. Lecture Notes in Computer Science, vol. 13177, pp. 142–161. Springer, Cham, Switzerland, Virtual Event (Mar 8–11, 2022). https://doi.org/10.1007/978-3-030-97121-2_6
32. Fouotsa, T.B., Moriya, T., Petit, C.: M-SIDH and MD-SIDH: Countering SIDH attacks by masking information. In: Hazay, C., Stam, M. (eds.) Advances in Cryptology – EUROCRYPT 2023, Part V. Lecture Notes in Computer Science, vol. 14008, pp. 282–309. Springer, Cham, Switzerland, Lyon, France (Apr 23–27, 2023). https://doi.org/10.1007/978-3-031-30589-4_10

33. Fouotsa, T.B., Petit, C.: Sims: A simplification of sigamal. In: Post-Quantum Cryptography: 12th International Workshop, PQCrypto 2021, Daejeon, South Korea, July 20–22, 2021, Proceedings. p. 277–295. Springer-Verlag, Berlin, Heidelberg (2021). https://doi.org/10.1007/978-3-030-81293-5_15, https://doi.org/10.1007/978-3-030-81293-5_15
34. Haidar, C.A., Passelègue, A., Stehlé, D.: Efficient updatable public-key encryption from lattices. In: Guo, J., Steinfeld, R. (eds.) Advances in Cryptology – ASIACRYPT 2023, Part V. Lecture Notes in Computer Science, vol. 14442, pp. 342–373. Springer, Singapore, Singapore, Guangzhou, China (Dec 4–8, 2023). https://doi.org/10.1007/978-981-99-8733-7_11
35. Jana, P., Dutta, R.: UPKE and UKEM schemes from supersingular isogenies. Cryptology ePrint Archive, Paper 2025/1010 (2025), https://eprint.iacr.org/2025/1010
36. Jost, D., Maurer, U., Mularczyk, M.: Efficient ratcheting: Almost-optimal guarantees for secure messaging. In: Ishai, Y., Rijmen, V. (eds.) Advances in Cryptology – EUROCRYPT 2019, Part I. Lecture Notes in Computer Science, vol. 11476, pp. 159–188. Springer, Cham, Switzerland, Darmstadt, Germany (May 19–23, 2019). https://doi.org/10.1007/978-3-030-17653-2_6
37. Kani, E.: The number of curves of genus two with elliptic differentials. Walter de Gruyter, Berlin/New York Berlin, New York (1997)
38. Kohel, D., Lauter, K., Petit, C., Tignol, J.P.: On the quaternion-isogeny path problem. LMS Journal of Computation and Mathematics **17**(A), 418–432 (2014)
39. Love, J., Boneh, D.: Supersingular curves with small noninteger endomorphisms. Open Book Series **4**(1), 7–22 (2020)
40. Maino, L., Martindale, C., Panny, L., Pope, G., Wesolowski, B.: A direct key recovery attack on SIDH. In: Hazay, C., Stam, M. (eds.) Advances in Cryptology – EUROCRYPT 2023, Part V. Lecture Notes in Computer Science, vol. 14008, pp. 448–471. Springer, Cham, Switzerland, Lyon, France (Apr 23–27, 2023). https://doi.org/10.1007/978-3-031-30589-4_16
41. Manulis, M., Slamanig, D., Valbusa, F.: SoK: Updatable public-key encryption. Cryptology ePrint Archive, Paper 2026/583 (2026), https://eprint.iacr.org/2026/583
42. Moriya, T., Onuki, H., Takagi, T.: SiGamal: A supersingular isogeny-based PKE and its application to a PRF. In: Moriai, S., Wang, H. (eds.) Advances in Cryptology – ASIACRYPT 2020, Part II. Lecture Notes in Computer Science, vol. 12492, pp. 551–580. Springer, Cham, Switzerland, Daejeon, South Korea (Dec 7–11, 2020). https://doi.org/10.1007/978-3-030-64834-3_19
43. Nakagawa, K., Onuki, H.: QFESTA: Efficient algorithms and parameters for FESTA using quaternion algebras. In: Reyzin, L., Stebila, D. (eds.) Advances in Cryptology – CRYPTO 2024, Part V. Lecture Notes in Computer Science, vol. 14924, pp. 75–106. Springer, Cham, Switzerland, Santa Barbara, CA, USA (Aug 18–22, 2024). https://doi.org/10.1007/978-3-031-68388-6_4
44. Onuki, H., Aikawa, Y., Takagi, T.: The existence of cycles in the supersingular isogeny graphs used in sike. In: 2020 International Symposium on Information Theory and Its Applications (ISITA). pp. 358–362. IEEE (2020)
45. Page, A., Robert, D.: Introducing clapoti(s): Evaluating the isogeny class group action in polynomial time. Cryptology ePrint Archive, Paper 2023/1766 (2023), https://eprint.iacr.org/2023/1766
46. Patra, S., Dutta, R., Dey, J.: Compact fully asynchronous updatable public key encryption scheme from hamming quasi-cyclic cryptosystem. Cryptology ePrint Archive, Paper 2026/736 (2026), https://eprint.iacr.org/2026/736

47. Peikert, C.: He gives C-sieves on the CSIDH. In: Canteaut, A., Ishai, Y. (eds.) Advances in Cryptology – EUROCRYPT 2020, Part II. Lecture Notes in Computer Science, vol. 12106, pp. 463–492. Springer, Cham, Switzerland, Zagreb, Croatia (May 10–14, 2020). https://doi.org/10.1007/978-3-030-45724-2_16
48. Robert, D.: Evaluating isogenies in polylogarithmic time. Cryptology ePrint Archive, Report 2022/1068 (2022), https://eprint.iacr.org/2022/1068
49. Robert, D.: Breaking SIDH in polynomial time. In: Hazay, C., Stam, M. (eds.) Advances in Cryptology – EUROCRYPT 2023, Part V. Lecture Notes in Computer Science, vol. 14008, pp. 472–503. Springer, Cham, Switzerland, Lyon, France (Apr 23–27, 2023). https://doi.org/10.1007/978-3-031-30589-4_17
50. Vélu, J.: Isogénies entre courbes elliptiques. Comptes-Rendus de l'Académie des Sciences **273**, 238–241 (1971)
51. Zanon, G.H.M., Simplicio, M.A., Pereira, G.C.C.F., Doliskani, J., Barreto, P.S.L.M.: Faster key compression for isogeny-based cryptosystems. IEEE Transactions on Computers **68**(5), 688–701 (2019). https://doi.org/10.1109/TC.2018.2878829

A Appendix

We recall a few basic definitions for public-key encryption (PKE) and updatable public-key encryption (UPKE).

Definition 17. *A public-key encryption scheme (PKE) is a tuple* (KeyGen, Enc, Dec) *of PPT algorithms satisfying the following :*

- $\mathsf{KeyGen}(1^\lambda) \to (\mathsf{sk}, \mathsf{pk})$*: Using the security parameter* λ*, the* KeyGen *algorithm generates a pair of secret-public key* $(\mathsf{sk}, \mathsf{pk})$*;*
- $\mathsf{Enc}(\mathsf{pk}, \mathsf{mex}) \to \mathsf{ctxt}$*: The* Enc *algorithm takes as input a message* $\mathsf{mex} \in \mathcal{M}$ *and a public key* pk*, and returns a ciphertext* ctxt*;*
- $\mathsf{Dec}(\mathsf{sk}, \mathsf{ctxt}) \to \mathsf{mex}$ *or* $\perp$*: On input the secret key* sk *and the ciphertext* ctxt*,* Dec *outputs the corresponding plaintext* mex *or* $\perp$ *in case of failure.*

Definition 18 (PKE Correctness [35]). *A public-key encryption* (PKE) *is correct if for all security parameter* λ *and all secret-public key pairs* $(\mathsf{pk}, \mathsf{sk})$ *generated by* KeyGen *algorithm, and all* $\mathsf{mex} \in \mathcal{M}$*, the following condition is satisfied:*

$$\mathsf{Dec}(\mathsf{sk}, \mathsf{Enc}(\mathsf{pk}, \mathsf{mex})) = \mathsf{mex} \tag{2}$$

Definition 19 (OW-CPA security). *Let* PKE = (KeyGen, Enc, Dec) *be a public key encryption scheme. The OW-CPA security of the PKE scheme is defined by a game between an adversary* $\mathcal{A}$ *and a challenger* $\mathcal{C}$ *defined in Fig. 7 . $\mathcal{A}$'s advantage is defined by :*

$$Adv_{\mathsf{PKE}}^{OW\text{-}CPA}(\mathcal{A}) = Pr[OW\text{-}CPA_{\mathsf{PKE}}^{\mathcal{A}} = 1]. \tag{3}$$

A PKE scheme is OW-CPA secure if for all PPT adversaries $\mathcal{A}$*, its advantage* $\mathsf{Adv}_{\mathsf{PKE}}^{\mathsf{OW\text{-}CPA}}(\mathcal{A})$ *is negligible.*

$\mathsf{OW\text{-}CPA}_{\mathsf{PKE}}^{\mathcal{A}}$

1 : $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$
2 : $\mathsf{mex}^* \leftarrow_\$ \mathcal{M}$
3 : $\mathsf{ctxt}^* \leftarrow \mathsf{Enc}(\mathsf{pk}, \mathsf{mex}^*)$
4 : $\mathsf{mex}' \leftarrow \mathcal{A}(\mathsf{ctxt}^*, \mathsf{pk})$
5 : **return** $(\mathsf{mex}' = \mathsf{mex}^*)$

Fig. 7: OW-CPA-PKE game

Definition 20 (UPKE Correctness [27]). *A UPKE scheme* (KeyGen, Enc, Dec, UpdPk, UpdSk) *with message space* $\mathcal{M}$ *is correct if:*

– $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *is a correct PKE with message space* $\mathcal{M}$*;*
– *for any* $t \in \mathbb{N}$*, any* $(\mathsf{sk}_0, \mathsf{pk}_0) \leftarrow \mathsf{KeyGen}(1^\lambda)$*, any* $(\mathsf{pk}_{i+1}, \mathsf{up}_i) \leftarrow \mathsf{UpdPk}(\mathsf{pk}_i, \rho)$*,* $\mathsf{sk}_{i+1} \leftarrow \mathsf{UpdSk}(\mathsf{sk}_i, \mathsf{up}_i)$ *for any* $i \in [0, t)$ *and any message* $m \in \mathcal{M}$*, we have*

$$\Pr[\mathsf{Dec}(\mathsf{sk}_{i+1}, \mathsf{Enc}(\mathsf{pk}_{i+1}, m)) = m] \geq 1 - \mathrm{negl}$$

where the probability is taken over the randomness of Enc*.*

Definition 21 (γ-spread). *A UPKE scheme* $\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{UpdPk}, \mathsf{UpdSk})$ *is* γ*-spread if for all* $\lambda, \ell \in \mathbb{N}$*, all* $(\mathsf{pk}_0, \mathsf{sk}_0) \leftarrow \mathsf{Gen}(1^\lambda)$*, all* $\mathsf{mex} \in \mathcal{M}$*, and all* $i \in \{0, \ldots, \ell\}$*, we have:*

$$-\log\left(\max_{\mathsf{ctxt} \in \{0,1\}^*} \Pr\left[\mathsf{ctxt} = \mathsf{Enc}(\mathsf{pk}_i, \mathsf{mex})\right]\right) \geq \gamma,$$

where $(\mathsf{pk}_j, \mathsf{up}_j) \leftarrow \mathsf{UpdPk}(\mathsf{pk}_{j-1})$*,* $\mathsf{sk}_j \leftarrow \mathsf{UpdSk}(\mathsf{sk}_{j-1}, (\mathsf{pk}_j, \mathsf{up}_j))$ *for* $j = 1, \ldots, \ell$*, and the probability is taken over the randomness used in* Enc*.*

This property aims to ensure that, given a message, the ciphertexts output over some encryption randomness are well spread. In other words, it characterizes how uniformly distributed are the ciphertexts, given messages and randomness.

Definition 22 (IND-CU-CCA security). *The IND-CU-CCA security of a UPKE scheme is defined by a game between an adversary* $\mathcal{A}$ *and a challenger* $\mathcal{C}$ *defined in Figure 8.* $\mathcal{A}$*'s advantage is defined by:*

$$\mathsf{Adv}^{\mathsf{IND\text{-}CU\text{-}CCA}}_{\Pi}(\mathcal{A}) = |\Pr[\mathsf{IND\text{-}CU\text{-}CCA}^{\mathcal{A}}_{\Pi} = 1] - \frac{1}{2}|. \tag{4}$$

A UPKE scheme is IND-CU-CCA secure if for all PPT adversaries $\mathcal{A}$*, its advantage* $\mathsf{Adv}^{\mathsf{IND\text{-}CU\text{-}CCA}}_{\Pi}(\mathcal{A})$ *is negligible.*

Proposition 23. *Given an OW-CR-CPA secure UPKE that is also* γ*-spread with* γ *at least* $\Omega(\lambda)$*, a symmetric encryption scheme that is OT-CPA secure and four random oracles, the transform of [6, §4.2] gives an IND-CU-CCA secure UPKE scheme.*

Definition 24 (OT-CPA Security). *The OT-CPA security of a symmetric-key encryption (SKE) scheme* Γ *is defined by a game between an adversary* $\mathcal{A}$ *and a challenger* $\mathcal{C}$*. The game is parameterized by the security parameter* λ *and proceeds as follows:*

1. $\mathcal{A}$ *sends a pair of messages* $(M^*_0, M^*_1) \in \mathcal{M}^2$ *to* $\mathcal{C}$*.*
2. $\mathcal{C}$ *chooses a random bit* $\mathsf{coin}^* \leftarrow_\$ \{0, 1\}$ *and a secret key* $k \leftarrow_\$ \mathsf{SK}$*.*
3. $\mathcal{C}$ *computes the ciphertext* $\mathsf{ct}^* \leftarrow \mathsf{SKE.Enc}(k, M^*_{\mathsf{coin}^*})$ *and sends* ct^* *to* $\mathcal{A}$*.*
4. *Finally,* $\mathcal{A}$ *outputs a guess* coin^d *for* coin^* *and the game ends.*

The adversary's advantage in this game is defined as:

$$\mathsf{Adv}^{\mathsf{OT\text{-}CPA}}_{\Gamma}(\mathcal{A}) = |\Pr[\mathsf{coin}^d = \mathsf{coin}^*] - \frac{1}{2}|.$$

If $\mathsf{Adv}^{\mathsf{OT\text{-}CPA}}_{\Gamma}(\mathcal{A})$ *is negligible in the security parameter* λ *for all probabilistic polynomial-time (PPT) adversaries* $\mathcal{A}$*, the SKE scheme* Γ *is said to satisfy OT-CPA security.*

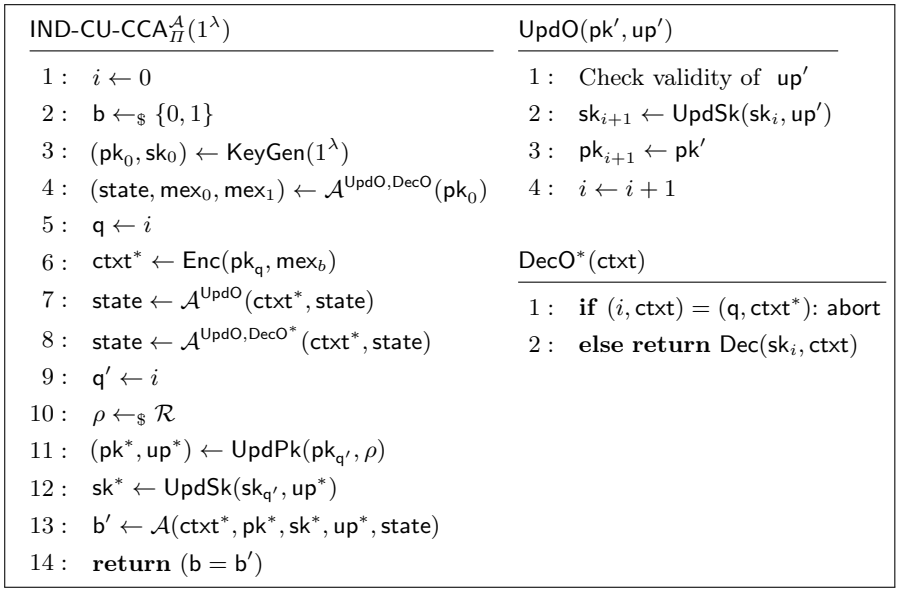

Fig. 8: IND-CU-CCA game for a UPKE scheme Π with chosen-update and decryption oracles.