# SIGNITC: Supersingular Isogeny Graph Non-Interactive Timed Commitments

Knud Ahrens
University of Passau, Germany
knud.ahrens@uni-passau.de

**Abstract**

Non-Interactive Timed Commitment schemes (NITC) allow to open any commitment after a specified delay $t_{\mathrm{fd}}$. This is useful for sealed bid auctions and as primitive for more complex protocols. We present the first NITC without repeated squaring or theoretical black box algorithms like NIZK proofs or one-way functions. It has fast verification, almost arbitrary delay and satisfies IND-CCA hiding and perfect binding. Our protocol is based on isogenies between supersingular elliptic curves making it presumably quantum secure, and all algorithms have been implemented as part of SQISign or other well-known isogeny-based cryptosystems. Additionally, it needs no trusted setup and can use known primes for SQISign or its higher dimensional variants.

**Keywords:** Non-interactive timed commitments, post-quantum, isogeny walks, Deuring correspondence.

## 1 Introduction

The concept of time-lock puzzles [37] has been around for more than twenty years and timed commitments [8] developed shortly after. We will use the rather new definition of Non-Interactive Timed Commitment schemes (NITC) by Katz, Loss, and Xu [31] from the year 2020. These protocols satisfy binding or non-malleability properties and efficient verification just like usual commitment schemes, but a commitment can be opened by anyone after some delay $t_{\mathrm{fd}}$. So hiding only lasts for this time $t_{\mathrm{fd}}$ and there are additional algorithms: one to verify that a commitment can be opened by others and another one to open the commitment forcefully in time at least $t_{\mathrm{fd}}$. A possible application is a sealed bid auction, where all bids can be revealed after time $t_{\mathrm{fd}}$ even if some of the bidders refuse to open their commitment. Other applications include e-voting, fair coin tossing or contract signing [8].

Our approach uses random walks in the isogeny graph of supersingular elliptic curves to construct a NITC, hence the name Supersingular Isogeny Graph Non-Interactive Timed Commitments or SIGNITC[1] for short. The main idea is that computing isogenies of large or non-smooth degree is slow, but if we know the endomorphism ring of the starting curve, we can find a smooth shortcut. So we use a secret isogeny to a curve with known endomorphism ring for fast

[1] pronounced like "signets"

commitment and verification, but the forced decommitment has to compute the delay isogeny and thus it needs time at least $t_{\mathrm{fd}}$.

The advantage of isogeny-based cryptography is that it is presumably quantum secure and relatively slow compared to other fields of post-quantum cryptography. Since we need a delay, this is a good thing. The field has undergone thorough scrutiny due to the candidates SIKE [30] and SQISign [22] in NIST competitions for post-quantum protocols and is still very active. The protocol only uses (known) isogeny-based cryptography, so we do not need to know several fields and this facilitates correct and secure implementations. This also means that we have no theoretical black box algorithms like zero-knowledge proofs, succinct non-interactive arguments (SNARGs) of knowledge or one-way functions. In addition, all needed calculations have already been implemented as subroutines in other cryptosystems. To our knowledge this is the first quantum secure NITC scheme with explicit algorithms. The only drawbacks are that some algorithms are still quite involved and that we need to differ slightly from the original definitions for hiding and binding.

**Related Work** Thyagarajan et al. [39] present an approach based on class groups using non-interactive zero-knowledge (NIZK) proofs. Katz et al. [31] and Chvojka and Jager [16] use protocols based on repeated squaring in a group of unknown order and NIZK proofs. Finally, Ambrona et al. [3] avoid NIZK proofs but still use repeated squaring. None of these is quantum secure.

NITC schemes are related to verifiable delay functions (VDF) [9] in the sense that both have fast verification and a function that needs a long time to evaluate. The main difference is the handling of secrets. For VDFs finding the correct response for a given challenge has to be slow for everyone. For NITC schemes however someone has to compute the commitment and therefore already knows the output of the slow task, namely finding the message to a given commitment. So we can construct NITC schemes from VDFs, but the contrary is difficult or impossible, depending on the protocol.

VDFs have direct applications to blockchains and there are already several approaches. Many are based on repeated squaring for the delay. A new publication [7] suggests that this might not be sequential. So contrary to current belief, repeated squaring could be parallelizable, disqualifying it as a delay function. Additionally, this is not quantum secure. There are even some isogeny-based candidates for VDFs, but they all still have some flaws. The pairing-based approach [21] is not quantum secure. Chavez-Saab et al. [14] use SNARGs and their verification time increases for larger delays. Finally, there is one base on Kani's criterion for abelian surfaces [24], but the authors state that it is not clear how to implement it. A different approach based on endomorphism rings [2] has the problem that the generation of a challenge also gives (a significant advantage in finding) the response. So it is closer to a NITC scheme and gave the initial idea for this article. Burdges and De Feo [11] introduced isogeny-based delay encryption, but they use the same delay as the pairing-based VDF.

**Structure of this Article** The remainder of this paper is structured as follows. First we give a definition of NITC schemes and discuss their properties. Next we recall the necessary definitions and fix the notations of isogeny-based cryptography. Readers familiar with one of these topics can briefly skim through the respective sections as we aimed to use standard notations. The sole difference is a slight variation in Definitions 5.11 and 5.14 of IND-CCA and BND-CCA security games. In Section 4 we present our protocol in full detail. Its security and its properties are discussed in Section 5. Finally, we give a short conclusion and outlook.

## 2 Non-Interactive Timed Commitments

In this section we recall NITC schemes and their properties. In their paper Katz et al. [31] gave the first formal definition of this concept.

**Definition 2.1** (NITC [31])**.** *A* $(t_{\text{com}}, t_{\text{cv}}, t_{\text{dv}}, t_{\text{fd}})$ non-interactive timed commitment scheme (NITC) *is a tuple* $\texttt{TC} = (\texttt{PGen}, \texttt{Com}, \texttt{ComVrfy}, \texttt{DecVrfy}, \texttt{FDecom})$ *of five algorithms with the following behavior:*

- *The randomized parameter generation algorithm* `PGen` *takes as input the security parameter* $1^\kappa$ *and outputs a common reference string* **crs**.
- *The randomized commit algorithm* `Com` *takes as input a string* **crs** *and a message* $m$*. It outputs a commitment* $\mathbf{C}$ *and proofs* $\pi_{\text{com}}$*,* $\pi_{\text{dec}}$ *in time at most* $t_{\text{com}}$.
- *The deterministic commitment verification algorithm* `ComVrfy` *takes as input a string* **crs***, a commitment* $\mathbf{C}$ *and a proof* $\pi_{\text{com}}$*. It outputs* **accept** *(if* $\mathbf{C}$ *could be forcefully decommitted) or* **reject** *in time at most* $t_{\text{cv}}$.
- *The deterministic decommitment verification algorithm* `DecVrfy` *takes as input a string* **crs***, a commitment* $\mathbf{C}$*, a message* $m$ *and a proof* $\pi_{\text{dec}}$*. It outputs* **accept** *or* **reject** *in time at most* $t_{\text{dv}}$.
- *The deterministic forced decommitment algorithm* `FDecom` *takes as input a string* **crs** *and a commitment* $\mathbf{C}$*. It outputs a message* $m$ *or* **invalid** *in time* at least $t_{\text{fd}}$.

*We require that for all* $\kappa$*, all* **crs** *output by* $\texttt{PGen}(1^\kappa)$*, all* $m$ *and all* $\mathbf{C}, \pi_{\text{com}}, \pi_{\text{dec}}$ *output by* $\texttt{Com}(\mathbf{crs}, m)$*, it holds that*

$$\texttt{ComVrfy}(\mathbf{crs}, \mathbf{C}, \pi_{\text{com}}) = \mathbf{accept} = \texttt{DecVrfy}(\mathbf{crs}, \mathbf{C}, m, \pi_{\text{dec}})$$

*and* $\texttt{FDecom}(\mathbf{crs}, \mathbf{C}) = m$.

To be relevant for applications a NITC also needs to satisfy three further properties. First we give a definition of *practicality* and then recall definitions for *hiding* and *binding* in our notation. There is no benefit in verification algorithms if we can verify faster by using `FDecom`. There are applications for protocols where `Com` is as slow as `FDecom`, but efficiently creating commitments even for long delays is desirable. Definition 2.2 tries to represent this. A more precise definition is difficult as one may wish for an absolute gap for small delays and relative gap for large delays.

**Definition 2.2** (Practicality)**.** *A NITC scheme is* practical*, if verification is much faster than forcefully opening the commitment, so* $t_{\text{cv}}, t_{\text{dv}} \ll t_{\text{fd}}$*. If in addition the commitment is also much faster than forced decommitment, i.e.* $t_{\text{com}} \ll t_{\text{fd}}$*, we call it* perfectly practical.

We present two IND-CCA security games and define hiding in terms of the probability that an adversary $\mathcal{A}$ wins the games. In both cases the adversary has access to an oracle for `FDecom` and a query is considered to have only a small computational cost. The first game is the one used by Katz et al. [31].

**Definition 2.3** (IND-CCA original [31])**.** *For a NITC scheme* $\mathtt{TC}$ *and an algorithm* $\mathcal{A}$*, define the game* $\mathtt{IND\text{-}CCA}^{\mathcal{A}}_{\mathtt{TC}}$ *as follows:*

1. *Compute* $\mathbf{crs} \leftarrow \mathtt{PGen}(1^\kappa)$*.*

2. *Run* $\mathcal{A}(\mathbf{crs})$ *in a preprocessing phase with access to* $\mathtt{FDecom}(\mathbf{crs}, \cdot)$*.*

3. *When* $\mathcal{A}$ *outputs* $(m_0, m_1)$*, choose a uniform bit* $b \leftarrow \{0,1\}$ *and then compute* $(\mathbf{C}_b, \pi_{\mathrm{com}}, \pi_{\mathrm{dec}}) \leftarrow \mathtt{Com}(\mathbf{crs}, m_b)$*. Give* $(\mathbf{C}_b, \pi_{\mathrm{com}})$ *to* $\mathcal{A}$*, who continues to have access to* $\mathtt{FDecom}(\mathbf{crs}, \cdot)$ *except that it may not query the oracle on the given commitment* $\mathbf{C}_b$*.*

4. *When* $\mathcal{A}$ *outputs a bit* $b'$*, it wins iff* $b' = b$*.*

The commitment $\mathbf{C}$ in our approach is a tuple and not a single value. Because of that we can only satisfy a slightly weaker variation of the IND-CCA security game. The new Definition 5.11 is given and discussed in Section 5.2. Hiding is defined with respect to an IND-CCA game. This allows us to evaluate the security of our NITC in terms of both the original and our adapted definition. Broadly speaking hiding guarantees that it is impossible to infer information about the message from the commitment. In our case hiding should hold at least for the time $t_{\mathrm{fd}}$ it takes to open a commitment by force, so for all $t_o < t_{\mathrm{fd}}$ in the following definition.

**Definition 2.4** (Hiding [31])**.** *A NITC scheme* $\mathtt{TC}$ *is* $(t_p, t_o, \varepsilon)$-CCA-secure *if for all adversaries* $\mathcal{A}$ *running in time at most* $t_p$ *in the preprocessing phase and time at most* $t_o$ *in the subsequent online phase,*

$$\Pr\left[\mathcal{A} \textit{ wins } \mathtt{IND\text{-}CCA}^{\mathcal{A}}_{\mathtt{TC}}\right] \leq \frac{1}{2} + \varepsilon.$$

Similar to hiding, binding is defined in terms of the probability that $\mathcal{A}$ wins a BND-CCA security game. Again, we have to modify it slightly and the new Definition 5.14 is given in Section 5.2.

**Definition 2.5** (BND-CCA original [31])**.** *For a NITC scheme* $\mathtt{TC}$ *and an algorithm* $\mathcal{A}$*, define the game* $\mathtt{BND\text{-}CCA}^{\mathcal{A}}_{\mathtt{TC}}$ *as follows:*

1. *Compute* $\mathbf{crs} \leftarrow \mathtt{PGen}(1^\kappa)$*.*

2. *Run* $\mathcal{A}(\mathbf{crs})$ *with access to* $\mathtt{FDecom}(\mathbf{crs}, \cdot)$*.*

3. $\mathcal{A}$ *outputs* $(m, \mathbf{C}, \pi_{\mathrm{com}}, \pi_{\mathrm{dec}}, m', \pi'_{\mathrm{dec}})$ *and wins iff* $\mathtt{ComVrfy}(\mathbf{crs}, \mathbf{C}, \pi_{\mathrm{com}}) =$ **accept** *and either:*

   - $m \neq m'$*, yet* $\mathtt{DecVrfy}(\mathbf{crs}, \mathbf{C}, m, \pi_{\mathrm{dec}})$ *and* $\mathtt{DecVrfy}(\mathbf{crs}, \mathbf{C}, m', \pi'_{\mathrm{dec}})$ *both output* **accept***;*
   - $\mathtt{DecVrfy}(\mathbf{crs}, \mathbf{C}, m, \pi_{\mathrm{dec}}) =$ **accept** *but* $\mathtt{FDecom}(\mathbf{crs}, \mathbf{C}) \neq m$*.*

Binding makes sure that a commitment can not be opened to two different messages and that `FDecom` gives the correct messages for valid commitments.

**Definition 2.6** (Binding [31]). *A NITC scheme* `TC` *is* $(t, \varepsilon)$-BND-CCA-secure *if for all adversaries $\mathcal{A}$ running in time $t$,*

$$\Pr\left[\mathcal{A} \text{ wins } \texttt{BND-CCA}_{\texttt{TC}}^{\mathcal{A}}\right] \leq \varepsilon.$$

# 3 Isogeny-based Cryptography

In this section we provide the necessary basics for isogeny-based cryptography, quaternion algebras and the Deuring correspondence. We also discuss some computational problems in this area.

## 3.1 Elliptic Curves and the Quaternion Algebra

Elliptic curves have ties to different fields resulting in several equivalent definitions. We will mostly follow the notation of Silverman [38], but restrict ourselves to aspects relevant for this paper.

**Definition 3.1** (Elliptic Curve). *An* elliptic curve *is a pair $(E, \infty)$, where $E$ is a curve of genus one and $\infty \in E$. It is* defined over a field $K$, *if it is defined over $K$ as a curve and $\infty \in E(K)$.*

We can define an addition of points on the curve making $(E, +)$ an additive group where $\infty$ is the neutral element. This permits scalar multiplication written as $[m]\colon E \to E$ and torsion subgroups $E[m] := \{P \in E \mid [m]P = \infty\}$.

**Definition 3.2** (Isogeny). *Let $E$ and $E'$ be elliptic curves. Then a morphism $\varphi\colon E \to E'$ such that $\varphi(\infty) = \infty$ is called an* isogeny. *If a non-zero isogeny $\varphi\colon E \to E'$ exists, then $E$ and $E'$ are called* isogenous.

In fact, every isogeny between two curves is also a group homomorphism. The isogenies from a curve $E$ into itself form the endomorphism ring $\operatorname{End} E$. Isogenies can be written as rational maps and their degree is defined by this map. Thus, the degree $\deg(\varphi \circ \varphi') = \deg \varphi \deg \varphi'$ is multiplicative. In addition, each isogeny $\varphi\colon E \to E'$ has a unique dual isogeny $\hat{\varphi}\colon E' \to E$ such that the composition $\hat{\varphi} \circ \varphi = [\deg \varphi]$ is the multiplication by the degree. The isogenies of degree 1 are the isomorphisms, and each isomorphism class can be labelled by the so-called $j$-invariant. This allows to construct the $\ell$-isogeny graph that has those $j$-invariants as vertices and isogenies of degree $\ell$ as edges.

**Definition 3.3** (Supersingularity). *Let $K$ be a field of characteristic $p > 0$ and $E$ an elliptic curve defined over $K$. The curve $E$ is* supersingular *if the torsion group $E[p]$ is trivial. Equivalently, this means that the endomorphism ring* $\operatorname{End} E$ *is an order in a quaternion algebra.*

For the rest of this paper $p > 3$ will be a large prime. This allows us to write every elliptic curve in short Weierstraß form as $E\colon y^2 = x^3 + Ax + B$ with $j(E) = 108(4A)^3/(4A^3 + 27B^2)$. For supersingular curves there is always a representation with $A, B, j$ in $\mathbb{F}_{p^2}$. The Galois conjugate curve $E^p\colon y^2 = x^3 + A^p x + B^p$ has $j(E^p) = j^p$. There are only $\lfloor p/12 \rfloor + \varepsilon$ supersingular elliptic curves for fields with characteristic $p$ where $\varepsilon \in \{0, 1, 2\}$. Hence, the subset $J_{SS} \subset \mathbb{F}_{p^2}$ of supersingular $j$-invariants has cardinality at least $\lfloor p/12 \rfloor$.

We have already seen in Definition 3.3 that supersingular curves are related to quaternion algebras. We are interested in the quaternion algebra $\mathcal{B}_{p,\infty}$ ramified at $p$ and infinity with $\mathbb{Q}$-basis $\{1, \mathrm{i}, \mathrm{j}, \mathrm{k}\}$ such that

$$\mathrm{i}^2 = -1, \quad \mathrm{j}^2 = -p, \quad \mathrm{k} = \mathrm{ij} = -\mathrm{ji}.$$

The (reduced) norm of an element $\alpha = a_0 + a_1\mathrm{i} + a_2\mathrm{j} + a_3\mathrm{k} \in \mathcal{B}_{p,\infty}$ is given by $\mathrm{nrd}(\alpha) = \alpha\bar{\alpha}$ for $\bar{\alpha} = a_0 - a_1\mathrm{i} - a_2\mathrm{j} - a_3\mathrm{k}$. The bilinear form $f(\alpha, \beta) = (\alpha\bar{\beta} + \beta\bar{\alpha})/2$ satisfies $f(\alpha, \alpha) = \alpha\bar{\alpha} = \mathrm{nrd}(\alpha)$, and two elements $\alpha, \beta \in \mathcal{B}_{p,\infty}$ are called orthogonal if $f(\alpha, \beta) = 0$. An order in $\mathcal{B}_{p,\infty}$ is a lattice that is also a subring, and it is maximal if its discriminant equals $p$. Now an elliptic curve $E$ is supersingular if and only if $\operatorname{End} E$ is isomorphic to a maximal order $\mathcal{O}$ in $\mathcal{B}_{p,\infty}$.

**Theorem 3.4** (Deuring Correspondence [25]). *The isomorphism classes of supersingular elliptic curves correspond to the isomorphism classes of invertible left $\mathcal{O}$-ideals in the quaternion algebra, for a fixed maximal order $\mathcal{O}$.*

This so-called Deuring correspondence also gives us that an $\ell$-isogeny $\varphi$ starting at $E$ corresponds to a left ideal $I_\varphi$ of norm $\ell$ in $\mathcal{O} \cong \operatorname{End} E$, and the image curve has an endomorphism ring isomorphic to the right order $\mathcal{O}_R(I_\varphi) = \{\alpha \in \mathcal{B}_{p,\infty} \mid I_\varphi\alpha \subseteq I_\varphi\}$ of $I_\varphi$, see [40, Ch. 42] for more details.

## 3.2 Application to Cryptography

Many isogeny-based protocols rely on secret walks in isogeny graphs of supersingular elliptic curves. The fact that the endomorphism ring is non-commutative gives rise to presumably quantum secure protocols. Moreover, the graphs have fast mixing properties, meaning that we reach an almost uniform distribution on the graph after a short random walk [20].

Taking $n$ steps in the $\ell$-isogeny graph corresponds (up to isomorphism) to an isogeny $\varphi\colon E \to E'$ of degree $d = \ell^n$. For our purposes the degree of such isogenies will always be coprime to the characteristic $p$ of the field and the isogeny $\varphi$ is determined by a point $K$ of order $d$ on the staring curve $E$. This point generates the kernel of $\varphi$ and we write $E' \cong E/\langle K\rangle$. In this case the $d$-torsion group $E[d]$ has $d^2$ elements and can be generated by two points $P, Q$ of order $d$ on $E$. This allows us to efficiently choose and describe a random walk by two integers $a, b$ such that $K = [a]P + [b]Q$. We can even use this to define a random walk starting on a different curve. For an isogeny $\psi\colon E \to E''$ with degree coprime to the degree of $\varphi$ the pushforward $[\psi]_\star\varphi$ is determined by the kernel $\langle\psi(K) = [a]\psi(P) + [b]\psi(Q)\rangle$ and starts at the codomain $E''$ of $\psi$. Note that although every supersingular elliptic curve has a representation in $\mathbb{F}_{p^2}$, the kernel of an isogeny and hence its generators might be elements of extensions $\mathbb{F}_{p^{2e}}$.

Now we list some computational tasks that are relevant for isogeny-based cryptosystems. First we present tasks that can be solved efficiently and have a polynomial or even constant complexity.

**Task 1:** Compute isogenies given their kernels.

**Task 2:** Given two elliptic curves $E, E'$, an isogeny $\varphi\colon E \to E'$ as well as the corresponding order $\mathcal{O} \cong \operatorname{End} E$ and ideal $I_\varphi$, compute $\mathcal{O}' \cong \operatorname{End} E'$.

**Task 3:** Given two elliptic curves $E, E'$, and the corresponding orders $\mathcal{O} \cong \operatorname{End} E$, $\mathcal{O}' \cong \operatorname{End} E'$, compute a connecting ideal $I$ corresponding to an isogeny $\varphi_I \colon E \to E'$.

**Task 4:** Given a left ideal $I$ of a maximal order $\mathcal{O} \subset \mathcal{B}_{p,\infty}$, find an equivalent ideal $J = I\upbeta$ (for $\upbeta \in \mathcal{B}^*_{p,\infty}$) with specific norm (usually small or smooth).

**Task 5:** Given $\mathcal{O} \cong \operatorname{End} E$, translate between isogenies $\varphi \colon E \to E'$ and their corresponding left $\mathcal{O}$-ideals $I_\varphi$.

Depending on the degree, Task 1 can be solved using Vélu's formulae [41] or the $\sqrt{\text{élu}}$ algorithm [6]. For Task 2 we can compute $\mathcal{O}'$ as $\mathcal{O}_R(I_\varphi)$ and the connecting ideal $I$ in Task 3 satisfies $\mathcal{O} = \mathcal{O}_L(I)$, where the left order $\mathcal{O}_L(I)$ is defined analogously to the right order $\mathcal{O}_R(I) = \mathcal{O}'$. Task 4 is solved by the KLPT algorithm [32] and Task 5 is addressed by subroutines of SQISign [22]. The right orders of the equivalent ideals from Task 4 only agree up to isomorphism, and if we translate them with Task 5 the codomains of the resulting isogenies only agree up to Galois conjugacy.

Note that these are not very specific and might have significantly different running times for special cases or when given additional information. For example, Task 1 is more efficient for smooth degrees than for non-smooth ones of similar size (see Section 5.1) and Task 5 can be done faster when given extra information.

In general, Task 5 only requires corresponding generators $\mathcal{O} = \langle \upalpha_0, \dots, \upalpha_3 \rangle \cong \langle \alpha_0, \dots, \alpha_3 \rangle = \operatorname{End} E$. Given an ideal $I$, the kernel of the corresponding isogeny $\varphi_I$ is the set of points $K \in E$ such that $\alpha(K) = \infty$ for all $\alpha \in \operatorname{End} E$ corresponding to an element of $I$. Given an isogeny $\varphi \colon E \to E'$ with kernel $\langle K \rangle$, the corresponding ideal $I_\varphi$ is the set of elements $\upalpha \in \mathcal{O}$ such that the corresponding $\alpha \in \operatorname{End} E$ satisfies $\alpha(K) = \infty$. If we know the norm or degree $d$ (coprime to the characteristic $p$) of the ideal or isogeny, we can (pre)compute the action of the generators $(\alpha_0, \dots, \alpha_3)$ of $\operatorname{End} E$ on the torsion group $E[d]$ and write it as $2 \times 2$ matrices $(A_0, \dots, A_3)$ with respect to a basis $(P, Q)$ of $E[d]$. Also, it suffices to find one point or one quaternion to generate the kernel or the ideal, respectively. Finding a generator is then reduced to finding a solution to a system of linear equations modulo $d$. If we additionally know two non-trivial endomorphisms $\theta$, $\eta$ such that $\langle K, \theta(K) \rangle = E[d]$ and the corresponding quaternions $\uptheta$, $\upeta$ are orthogonal, we can solve $[a]K + [b]\theta(K) = \eta(K)$ to get $I_\varphi = \mathcal{O}\upalpha + \mathcal{O}d$ for $\upalpha = a + b\uptheta - \upeta$. This can be done by writing $K = [s]P + [t]Q \in E[d]$ as a vector $(s, t)^\top$ and $\theta, \eta$ as matrices in terms of $(A_0, \dots, A_3)$ in order to solve this as a matrix equation as in [15, Algorithm 23].

To create a delay we need moderately hard problems, which are still polynomial in complexity but might take a considerable time to compute. In Section 5.1 we show that Task 1 can be made sufficiently slow. The following hard problems have a conjectured exponential complexity (see Section 5.1) and are equivalent [42]. They are the basis for encryption or signature schemes like CSIDH [13] or SQISign [22]. In our case they ensure that there are no shortcuts for the forced decommitment.

**Problem 3.5** (Isogeny Path Problem)**.** *Given two (isogenous) supersingular elliptic curves $E, E'$ and a prime $\ell$, find a path from $E$ to $E'$ in the $\ell$-isogeny graph.*

**Problem 3.6** (Endomorphism Ring Problem). *Given a supersingular elliptic curve $E$, find four endomorphisms that generate* $\operatorname{End} E$ *as a lattice.*

**Problem 3.7** (Maximal Order Problem). *Given a supersingular elliptic curve $E$, find four quaternions in $\mathcal{B}_{p,\infty}$ that generate a maximal order $\mathcal{O} \cong \operatorname{End} E$.*

**Remark 3.8.** *Knowledge of endomorphism rings can break the hard problems. If we know both endomorphism rings the first hard problem becomes polynomial using Tasks 3 - 5. If we know an isogeny from a curve with known endomorphism ring to our curve the third hard problem becomes polynomial by Task 2. Using Task 5 a solution for the third problem can be translated into a solution for the second hard problem.*

Finding supersingular elliptic curves can basically be done in two ways. We can reduce an elliptic curve in characteristic 0 modulo a prime and check if the resulting curve is supersingular, or take a random isogeny starting at one of these curves. In both cases the endomorphism ring of the final curve can be computed either via reduction or by transport along the isogeny. But as discussed in Remark 3.8 this weakens the hard problems. Hence, many cryptosystems require curves with unknown endomorphism ring. This in turn forces them to use a multi-party computation or a trusted authority in their setup to ensure that no single participant knows a complete path from a curve with known endomorphism ring to the one used. See [4] for more information. Note that the present cryptosystem has the advantage of not relying on a curve with unknown endomorphism ring.

# 4 The Protocol

Now we can combine the previous two sections and present our construction. First we give a high-level overview and discuss some challenges. Then we look at the algorithms and choices for the parameters.

## 4.1 Overview

At the heart of our protocol is an isogeny $\varphi_T$ of degree $d_T$, which takes time $t_{\text{fd}}$ to evaluate and hence causes the delay. Its domain is a public supersingular elliptic curve $E_s$ with secret $\mathcal{O}_s \cong \operatorname{End} E_s$ and its kernel is generated by a publicly known point $K_T$ on $E_s$. We use the $j$-invariant $j_T$ of the codomain $E_T$ of $\varphi_T$ to hide the message $m \in M$. Therefore, an adversary needs to compute $E_T$ (or rather $j_T = j(E_T)$) in order to break hiding or to open the commitment by force. We can choose how long the commitment should be kept secret by setting the degree $d_T$ accordingly. This gives us hiding. Since $E_s$ and $K_T$ are part of the commitment, the codomain $E_T \cong E_s/\langle K_T \rangle$ is fixed (up to isomorphism) and we have perfect binding.

For verification to be faster than forced opening, we need a more efficient way to compute $j_T$. The starting curve $E_0$ has a known endomorphism ring, which allows us to compute elements of $\operatorname{End} E_0 \cong \mathcal{O}_0$ efficiently using precomputations. During the commitment we choose a smooth secret isogeny $\varphi_s \colon E_0 \to E_s$ and an isogeny $\varphi'_T \colon E_0 \to E'_T$ of degree $d_T$. We set the delay isogeny $\varphi_T = [\varphi_s]_\star \varphi'_T$, the composition $\psi = \varphi_T \circ \varphi_s$ and use the precomputations to find the corresponding ideals $I_s, I'_T, I_T, I_\psi$. Now we translate these ideals into isogenies that are efficiently computable using so-called `IdealToIsogeny` algorithms. Note that we can also use higher dimensional isogenies for this step. For the 1-dimensional variant we compute another isogeny $\widetilde{\psi}\colon E_0 \to E_T$ of smooth degree (Tasks 4 & 5 from Section 3.2). This is visualized in Figure 1.

We give $\varphi_s$ and $\varphi'_T$ to the verifier as part of the decommitment proof, so `Com` and `DecVrfy` can compute $E_T$ as the codomain of $\widetilde{\psi}$ (or the result of the higher dimensional `IdealToIsogeny` algorithm)[2]. An adversary only knows $E_s$, but not $\varphi_s$ or $\varphi'_T$ and hence can neither compute $\mathcal{O}_s \cong \operatorname{End} E_s$ nor the ideals $I_s, I'_T, I_T, I_\psi$. Therefore, it has no efficient way to compute shortcuts. This gives us the preferred difference in speed between verification and forced opening.

$$\begin{tikzcd}
E_0 \arrow[rr, "\varphi'_T"] \arrow[d, "\varphi_s"'] \arrow[drr, "\widetilde{\psi}"] \arrow[drr, bend right, "\psi = \varphi_T \circ \varphi_s"'] & & E'_T \cong E_0/\langle K'_T\rangle \arrow[d, dotted, "\varphi'_s = [\varphi'_T]_\star \varphi_s"] \\
E_s \cong E_0/\langle K_s\rangle \arrow[rr, "\varphi_T = [\varphi_s]_\star \varphi'_T"'] & & E_T \cong E_s/\langle K_T\rangle
\end{tikzcd}$$

Figure 1: Walk in the isogeny graph with (efficiently computable) smooth degrees $\deg(\varphi_s), \deg(\widetilde{\psi})$ and large and/or non-smooth degree $\deg \varphi_T$.

To efficiently verify the validity of a commitment, we need to map the $j$-invariant $j_T$ into the group of messages $M$. This map has to satisfy the following property. Otherwise the commitment might leak information about $j_T$.

**Definition 4.1** (Inverse Resistant Functions)**.** *A function $f\colon X \to Y$ is $\lambda$-*inverse resistant*, if for all $y \in Y$ the preimage $f^{-1}(y) \subseteq X$ has at least $2^\lambda$ elements.*

This definition is other than one-way functions, since finding an element in the preimage is allowed as long as the probability to find the original input is sufficiently small. It also differs from hash functions, which are mostly considered to be collision resistant. A simple projection with a sufficiently large preimage set satisfies this definition but is neither a one-way function nor a proper hash function.

## 4.2 Algorithms

As seen in Definition 2.1 we have five algorithms `PGen`, `Com`, `ComVrfy`, `DecVrfy` and `FDecom`. In this subsection we give pseudocode for each algorithm and discuss their (relative) speed and some subroutines. The `IdealToIsogeny` algorithms used in `Com` and `DecVrfy` are discussed in Section 4.2.6.

[2] In practice, they only compute $\widetilde{E}_T$ isomorphic to $E_T$ or its Galois conjugate $E_T^p$.

### 4.2.1 Parameter Generation

The parameter generation `PGen` defines the security of the whole protocol and fixes the delay $t_{\mathrm{fd}}$. It sets all general parameters like the characteristic $p$ of the finite fields, the starting curve $E_0$, $\mathcal{O}_0 \cong \operatorname{End} E_0$, the degrees $d_s$ and $d_T$, as well as the message group $(M, \oplus)$ and the inverse resistant function $F\colon J_{SS} \to M$. It also provides some precomputations that improve the speed of the commitment and the decommitment verification. These precomputations include bases of the $d_s$- and $d_T$-torsion groups of $E_0$, and matrices that correspond to the action of two endomorphisms $\theta$ and $\eta$ on $E_0[d_s]$ and $E_0[d_T]$. The quaternions corresponding to $\theta$ and $\eta$ are orthogonal and $\langle R, \theta(R)\rangle = E_0[d]$ for all $R \in E_0[d]$ of maximal order. This permits efficient translation of isogenies into ideals as in [15, Algorithm 23]. It may also include additional information to improve the translation of these ideals into suitable isogenies. For the 1-dimensional approach this can be a smooth integer $d_t \approx p^3$ and matrices that correspond to the action of $\operatorname{End} E_0$ on $E_0[d_t]$. Its output is the common reference string $\mathbf{crs}$.

**Algorithm 1** Parameter generation algorithm `PGen`

```
Require: Security parameter $1^\kappa$
Ensure: $\mathbf{crs} = (\mathbf{crs}_0, \mathbf{crs}_s, \mathbf{crs}_T, \mathbf{crs}_{\mathtt{ItI}})$
Ensure: $\mathbf{crs}_0 = (p, E_0, \operatorname{End} E_0, \mathcal{O}_0, \theta, \eta, M, F)$
Ensure: $\mathbf{crs}_s = (d_s, P_s, Q_s, A_\theta, A_\eta)$, $\mathbf{crs}_T = (d_T, P'_T, Q'_T, e, B_\theta, B_\eta)$
Choose prime $p$ of right size
Choose supersingular elliptic curve $E_0$ with known $\mathcal{O}_0 \cong \operatorname{End} E_0$
Find corresponding bases $\mathcal{O}_0 = \langle \upalpha_0, \dots, \upalpha_3 \rangle$ and $\operatorname{End} E_0 = \langle \alpha_0, \dots, \alpha_3 \rangle$
Choose a group $(M, \oplus)$ with efficient membership testing as message space
Choose an efficient, inverse resistant function $F\colon J_{SS} \to M$ oblivious to
    Galois conjugacy, i.e. $F(j) = F(j^p)$
Choose smooth $d_s \in \mathbb{N}$ such that $E_0[d_s] \subseteq E_0(\mathbb{F}_{p^2})$
Find $P_s \in E_0[d_s]$ of maximal order $\operatorname{ord}(P_s) = d_s$
Choose $e, d_T \in \mathbb{N}$ such that $d_T$ is coprime to $d_s$ and $E_0[d_T] \subseteq E_0(\mathbb{F}_{p^{2e}})$
Find $P'_T \in E_0[d_T]$ of maximal order $\operatorname{ord}(P'_T) = d_T$

Find endomorphisms $\theta, \eta \in \operatorname{End} E_0$ such that the corresponding quaternions
    $\uptheta, \upeta$ are orthogonal and $\langle P_s, \theta(P_s)\rangle = E_0[d_s]$, $\langle P'_T, \theta(P'_T)\rangle = E_0[d_T]$
Set $\mathbf{crs}_0 = (p, E_0, \operatorname{End} E_0, \mathcal{O}_0, \theta, \eta, M, F)$, $Q_s = \theta(P_s)$ and $Q'_T = \theta(P'_T)$
Compute the action of $\theta, \eta$ on $E_0[d_s]$ as matrices $A_\theta, A_\eta \in \mathrm{Mat}_{2\times 2}(\mathbb{Z}/d_s\mathbb{Z})$
    with respect to the basis $(P_s, Q_s)$
Set $\mathbf{crs}_s = (d_s, P_s, Q_s, A_\theta, A_\eta)$
Compute the action of $\theta, \eta$ on $E_0[d_T]$ as matrices $B_\theta, B_\eta \in \mathrm{Mat}_{2\times 2}(\mathbb{Z}/d_T\mathbb{Z})$
    with respect to the basis $(P'_T, Q'_T)$
Set $\mathbf{crs}_T = (d_T, P'_T, Q'_T, e, B_\theta, B_\eta)$                    ▷ Fixes delay $t_{\mathrm{fd}}$
Compute $\mathbf{crs}_{\mathtt{ItI}}$            ▷ Depends on chosen IdealToIsogeny algorithm

return $\mathbf{crs} = (\mathbf{crs}_0, \mathbf{crs}_s, \mathbf{crs}_T, \mathbf{crs}_{\mathtt{ItI}})$
```

A detailed description can be found in Algorithm 1. The speed is dominated by finding generators of $E_0[d_T]$ and computing the action of $\theta$ and $\eta$. The bottlenecks are checking the order and linear independency of two points in

$E_0(\mathbb{F}_{p^{2e}})$ and decomposing the images of this basis under the endomorphisms in terms of this basis.

### 4.2.2 Commitment

The commitment algorithm `Com` takes as input a message $m \in M$ and outputs a tuple $(\mathbf{C}, \pi_{\text{com}}, \pi_{\text{dec}})$. First it chooses a random isogeny $\varphi_s \colon E_0 \to E_s$ of smooth degree $d_s$ and a second random isogeny $\varphi'_T \colon E_0 \to E'_T$ of large and/or non-smooth degree $d_T$. These can be extended to a SIDH-square with the pushforwards $\varphi_T = [\varphi_s]_\star \varphi'_T \colon E_s \to E_T$ and $\varphi'_s = [\varphi'_T]_\star \varphi_s \colon E'_T \to E_T$, see Figure 1. Let $j_T = j(E_T)$ denote the $j$-invariant of $E_T$. Then it computes $F(j_T)$ and $u = m \ominus F(j_T) \in M$.[3] The commitment itself $\mathbf{C} = (E_s, K_T, u)$ is again a tuple of a supersingular elliptic curve $E_s$, a point $K_T$ on $E_s$ that generates the kernel of $\varphi_T$ and $u \in M$. While the commitment proof $\pi_{\text{com}}$ is empty, the decommitment proof $\pi_{\text{dec}}$ allows to reconstruct the secret isogenies $\varphi_s$, $\varphi'_T$ and to use the same method for computing $F(j_T)$ as in the commitment algorithm. If `Com` uses heuristic algorithms (like KLPT) for finding special elements $\beta \in I$ or $\gamma \in \mathcal{O}_0$, i.e. equivalent ideals $J = I\beta$ or endomorphisms $\gamma \in \operatorname{End} E_0$, these can be added to $\pi_{\text{dec}}$ to make `DecVrfy` deterministic.

To compute $F(j_T)$ efficiently, we need a faster method than computing $\varphi_T$ directly. Since $\varphi_s$ and $\varphi'_T$ are isogenies with domain $E_0$ there are efficient ways to compute the corresponding ideals $I_s$ and $I'_T$. The ideal $I_\psi$ corresponding to $\psi = \varphi_T \circ \varphi_s \colon E_0 \to E_T$ can then be computed as $I_\psi = I_s I_T = I_s([I_s]_\star I'_T) = I_s \cap I'_T$ by Lemma 3 from SQISign [22]. Now we use one of the `IdealToIsogeny` algorithms discussed in Section 4.2.6 to efficiently compute a curve $\widetilde{E}_T$ isomorphic to $E_T$ or its Galois conjugate $E_T^p$ and $F(j_T) = F(j(\widetilde{E}_T))$.

The individual steps are given in Algorithm 2. In SQISign [15] the authors state that converting between ideals and isogenies is the bottleneck of their computation. Therefore, we assume that the slowest part of this algorithm is computing `IdealToIsogeny`. We use (efficient) `IdealToIsogeny` algorithms based on those of SQISign or their improvements using higher dimensions. So the commitment algorithm `Com` is efficient and for properly chosen $d_T$ it is faster than computing the delay isogeny $\varphi_T$ (over $\mathbb{F}_{p^{2e}}$). A more detailed discussion can be found in Section 5.3.

### 4.2.3 Commitment Verification

Algorithm 3 shows the commitment verification `ComVrfy`. It is fast since it only needs to check if the three parts of the commitment are of the correct form. Namely, $E_s$ is an elliptic curve, $K_T$ is a point on that curve and $u$ is an element of the group $M$. All of these membership tests can be done efficiently. If we want to assure that forced opening does not take too long, we can also check if $K_T \in \mathbb{F}^2_{p^{2e}}$. This sets an upper bound for the degree $d_T$ of $\varphi_T$ as $E_s(\mathbb{F}_{p^{2e}}) = E_s[p^e - (-1)^e]$ (if $|E_0(\mathbb{F}_{p^2})| = (p+1)^2$) and hence $d_T \mid p^e - (-1)^e$.

**Remark 4.2.** *If we want to make sure that forcefully opening* $\mathbf{C}$ *takes not much longer than* $t_{\text{fd}}$*, we can check* $\operatorname{ord} K_T \mid d_T$*. Remark 5.18 discusses how this affects the speed.*

[3] Here $\ominus$ means the addition of the inverse in the group $(M, \oplus)$ such that $m \ominus m$ is the neutral element in $M$ and in particular $u \oplus F(j) = m \ominus F(j) \oplus F(j) = m$.

**Algorithm 2** Commitment algorithm `Com`

```
Require: Common reference string $\mathbf{crs}$, message $m \in M$
Ensure: $(\mathbf{C}, \pi_{\text{com}}, \pi_{\text{dec}}) = \big((E_s, K_T, u), (), (s, t, \mathbf{r})\big)$
 1: Choose random $s \in [0, d_s)$ and compute $K_s = P_s + [s]Q_s \in E_0[d_s]$
 2: Compute $E_s \cong E_0/\langle K_s \rangle$ via Vélu's formulae
 3: Set $v = (1, s)^\top$ and $A = (v, A_\theta v) \in \mathrm{GL}_2(\mathbb{Z}/d_s\mathbb{Z})$
 4: Compute $(a_1, a_2)^\top = A^{-1}A_\eta v$ and ideal $I_s = \mathcal{O}_0(a_1 + a_2\theta - \eta) + \mathcal{O}_0 d_s$
    corresponding to isogeny $\varphi_s\colon E_0 \to E_s$ with kernel $\langle K_s \rangle$

 5: Choose random $t \in [0, d_s)$ and set $w = (1, t)^\top$    ▷ We use $d_s$ for efficiency
 6: Set $B = (w, B_\theta w) \in \mathrm{GL}_2(\mathbb{Z}/d_T\mathbb{Z})$ and compute $(b_1, b_2)^\top = B^{-1}B_\eta w$
 7: Compute ideal $I'_T = \mathcal{O}_0(b_1 + b_2\theta - \eta) + \mathcal{O}_0 d_T$ corresponding to isogeny
    $\varphi'_T\colon E_0 \to E'_T$ with kernel $\langle K'_T = P'_T + [t]Q'_T \rangle$
 8: Compute ideal $I_\psi = I_s \cap I'_T$ corresponding to isogeny $\psi = [\varphi_s]_\star \varphi'_T \circ \varphi_s$
 9: Use IdealToIsogeny to get $\widetilde{E}_T$ isomorphic to $E_T \cong E_s/\langle \varphi_s(K'_T) \rangle$ or $E_T^p$,
    and the random elements $\mathbf{r}$ used to find it (Go back to step 5 if it fails)

10: Compute $K_T = \varphi_s(P'_T + [t]Q'_T) \in E_s[d_T]$
11: Compute $\widetilde{j}_T = j(\widetilde{E}_T)$ and $u = m \ominus F(\widetilde{j}_T) \in M$    ▷ $F(\widetilde{j}_T) = F(j(E_T))$
12: Set $\mathbf{C} = (E_s, K_T, u)$    ▷ Commitment
13: Set $\pi_{\text{com}} = ()$    ▷ Commitment proof (empty)
14: Set $\pi_{\text{dec}} = (s, t, \mathbf{r})$    ▷ Decommitment proof
15: return $(\mathbf{C}, \pi_{\text{com}}, \pi_{\text{dec}})$
```

**Algorithm 3** Commitment verification algorithm `ComVrfy`

```
Require: Common reference string $\mathbf{crs}$, commitment $\mathbf{C}$ and proof $\pi_{\text{com}}$
 1: Check if $E_s$ is an elliptic curve over $\mathbb{F}_{p^2}$, $K_T \in E_s$ and $u \in M$
    Optional: check $K_T \in \mathbb{F}^2_{p^{2e}}$    ▷ Check upper bound for degree of $\varphi_T$
 2: return (accept/reject)
```

### 4.2.4 Decommitment Verification

The decommitment verification `DecVrfy` (Algorithm 4) is similar to the commitment algorithm. It first reconstructs $\varphi_s, \varphi'_T$ from $\pi_{\text{dec}}$ and verifies $\varphi_s\colon E_0 \to E_s$ and $K_T = \varphi_s(P'_T + [t]Q'_T)$. Then it computes the ideals $I_s$, $I'_T$ and $I_\psi = I_s \cap I'_T$ corresponding to $\varphi_s$, $\varphi'_T$ and $\psi = \varphi_T \circ \varphi_s$, respectively. Now it checks that the rest of $\pi_{\text{dec}}$ is of the expected form and uses it to run `IdealToIsogeny` as a deterministic algorithm to efficiently find $\widetilde{E}_T$. Finally, it computes $\widetilde{j}_T = j(\widetilde{E}_T)$ and checks if $u \oplus F(\widetilde{j}_T) = m$. As stated above, we assume the slowest part of this algorithm to be the computation of `IdealToIsogeny`. Again, this is still faster than forced decommitment (cf. Section 5.3).

### 4.2.5 Forced Decommitment

In terms of the number of tasks the forced decommitment algorithm is rather simple. It just computes $E_T$ as codomain of the isogeny $\varphi_T$ given by the point $K_T$ that generates its kernel. From there it recovers the message $m = u \oplus F(j(E_T))$. Computing an isogeny $\varphi_T$ can be made slow if its degree $d_T$ is

**Algorithm 4** Decommitment verification algorithm `DecVrfy`

```
Require: Common reference string crs, commitment C
Require: Message $m$, decommitment proof $\pi_{\text{dec}}$
Compute $K_s = P_s + [s]Q_s \in E_0[d_s]$ and check $E_s \cong E_0/\langle K_s \rangle$
Compute $K'_T = P'_T + [t]Q'_T \in E_0[d_T]$ and check $\varphi_s(K'_T) = K_T$

Set $v = (1, s)^\top$ and $A = (v, A_\theta v) \in \mathrm{GL}_2(\mathbb{Z}/d_s\mathbb{Z})$
Compute $(a_1, a_2)^\top = A^{-1}A_\eta v$ and ideal $I_s = \mathcal{O}_0(a_1 + a_2\theta - \eta) + \mathcal{O}_0 d_s$
    corresponding to isogeny $\varphi_s\colon E_0 \to E_s$ with kernel $\langle K_s \rangle$
Set $w = (1, t)^\top$ and $B = (w, B_\theta w) \in \mathrm{GL}_2(\mathbb{Z}/d_T\mathbb{Z})$
Compute $(b_1, b_2)^\top = B^{-1}B_\eta w$ and ideal $I'_T = \mathcal{O}_0(b_1 + b_2\theta - \eta) + \mathcal{O}_0 d_T$
    corresponding to isogeny $\varphi'_T\colon E_0 \to E'_T$ with kernel $\langle K'_T \rangle$
Compute ideal $I_\psi = I_s \cap I'_T$ corresponding to isogeny $\psi = [\varphi_s]_\star\varphi'_T \circ \varphi_s$
Check that the elements in r are of the correct form
Use r to recompute IdealToIsogeny as deterministic algorithm to get
    codomain $\widetilde{E}_T$ isomorphic to $E_T \cong E_s/\langle K_T \rangle$ or $E_T^p$

Compute $\widetilde{j}_T = j(\widetilde{E}_T)$ and check $u \oplus F(\widetilde{j}_T) = m$        ▷ $F(\widetilde{j}_T) = F(j(E_T))$
return (accept/reject)
```

sufficiently large and/or non-smooth (cf. Theorem 5.8), especially when the calculations have to be done in a field extension $\mathbb{F}_{p^{2e}}$. This allows us to make Algorithm 5 (almost) arbitrarily slow.

**Algorithm 5** Forced decommitment algorithm `FDecom`

```
Require: Common reference string crs, commitment C
Ensure: Message $m$
Compute $E_T \cong E_s/\langle K_T \rangle$ via Vélu's formulae or $\sqrt{\text{élu}}$ algorithm
Compute $j_T = j(E_T)$ and $m = u \oplus F(j_T)$
return $m$
```

### 4.2.6 Ideal to Isogeny Algorithms

Explicitly computing the Deuring correspondence (Task 5 in Section 3.2) and in particular translating ideals into isogenies is part of SQISign [22]. There have been several improvements of this step including the application of higher dimensional isogenies. All approaches sample random elements until they find ones with special properties. This might fail, but their respective publications provide heuristics for parameters such that they succeed with high probability.

Using 1-dimensional isogenies we can do something similar to SQISign. We use an algorithm like KLPT [32, 36] to find an equivalent ideal $\widetilde{I}_\psi \sim I_\psi$ of norm $\widetilde{d}_\psi \approx p^3$ a power of 2. Then we translate $\widetilde{I}_\psi$ into its corresponding isogeny $\widetilde{\psi}\colon E_s \to \widetilde{E}_T$ using the methods of SQIsign [15] with the improvements of [23]. This allows us to efficiently compute the $j$-invariant $\widetilde{j}_T = j(\widetilde{E}_T)$ such that $F(\widetilde{j}_T) = F(j_T)$. Note that SQISign uses ideals with norm of size $p^{15/4} > p^3 \approx \widetilde{d}_\psi$ and is still considered efficient. If we use Algorithm 6, we need SQISign-friendly primes, but $\mathbf{crs}_{\mathtt{ItI}}$ in Algorithm 1 can be empty.

**Algorithm 6** 1-dimensional `IdealToIsogeny` algorithm (SQIsign)

```
Require: Common reference string crs, left $\mathcal{O}_0$-ideal $I_\psi$
Ensure: $(\widetilde{E}_T, \mathbf{r}) = (\widetilde{E}_T, \beta_\psi)$
Compute equivalent ideal $\widetilde{I}_\psi = I_\psi \beta_\psi / \operatorname{nrd}(I)$ with smooth norm $\widetilde{d}_\psi \approx p^3$
    using KLPT-like alogorithms        ▷ Optional: require $\widetilde{d}_\psi \mid d_t$
Compute corresponding isogeny $\widetilde{\psi}\colon E_0 \to \widetilde{E}_T$ of degree $\widetilde{d}_\psi$ as in SQIsign
return $(\widetilde{E}_T, \beta_\psi)$
```

If we go to higher dimensional isogenies we can find efficiently computable representations of $\varphi_T$ or $\psi$, i.e. embeddings into higher dimensional isogenies. The degree of these representations is usually a power of 2 that divides $p+1$ and hence much smaller than $p^3$. This makes the higher dimensional variants more efficient. For example, `IdealToIsogeny` from SQIsign2D-West [5] allows to evaluate $\psi$ using $I_\psi$ and a chain of $(2,2)$-isogenies. The `IdealToIsogeny` algorithm from [35] can use $I_s$, $\varphi_s$ and $I_T = I_s^{-1} I_s I_T = I_s^{-1} I_\psi$ to find a representation of $\varphi_T$ via $(2,2)$-isogenies. There are other 2-dimensional `IdealToIsogeny` algorithms (e.g. [33]) and 4- or 8-dimensional `IdealToIsogeny` algorithms like in SQIsignHD [19]. For higher dimensional `IdealToIsogeny` algorithms the slowest part is usually evaluating the higher dimensional isogenies, but we can choose the degree $d_T$ of the delay isogeny $\varphi_T$ such that computing the higher dimensional isogeny is faster than computing $\varphi_T$. We present one approach based on SQIsign2D-West in Algorithm 7. It needs $\mathbf{crs}_{\mathtt{ItI}} = (g, P_g, Q_g)$ where the points $P_g, Q_g \in E_0(\mathbb{F}_{p^2})$ form a basis of the $2^g$-torsion of $E_0$.

**Algorithm 7** 2-dimensional `IdealToIsogeny` algorithm (SQIsign2D-West)

```
Require: Common reference string crs, left $\mathcal{O}_0$-ideal $I_\psi$
Ensure: $(\widetilde{E}_T, \mathbf{r}) = (\widetilde{E}_T, (\beta_1, \beta_2, a_1, a_2, f, \gamma_1, \gamma_2))$
Find equivalent ideals $I_1 = I_\psi \beta_1 / \operatorname{nrd}(I_\psi), I_2 = I_\psi \beta_2 / \operatorname{nrd}(I_\psi)$ of odd and
    coprime norms $d_1, d_2 \approx \sqrt{p}$ and $a_1, a_2 \in \mathbb{N}$ such that $a_1 d_1 + a_2 d_2 = 2^f$,
    $f \leq g$ and $\gcd(a_1 d_1, a_2 d_2) = 1$
Find $\gamma_1, \gamma_2 \in \operatorname{End} E_0$ of degrees $a_1(2^g - a_1)$ and $a_2(2^g - a_2)$, respectively,
    using FullRepresentInteger
Set $\mathbf{r} = (\beta_1, \beta_2, a_1, a_2, f, \gamma_1, \gamma_2)$
Construct $(2^g, 2^g)$-isogenies $\Phi_i\colon E_0 \times E_0 \to E_i \times E_i'$ with kernels
    $\langle([a_i]P_g, \gamma_i(P_g)), ([a_i]Q_g, \gamma_i(Q_g))\rangle$ for $i \in \{1, 2\}$
Compute $\varphi_i(P_g), \varphi_i(Q_g)$ for isogenies $\varphi_i\colon E_0 \to E_i$ of degrees $a_i$ as
    $\Phi_i(P, \infty) = (\varphi_i(P), *)$ for $i \in \{1, 2\}$
Compute $\gamma = \widehat{\psi_2} \circ \psi_1 \in \operatorname{End} E_0$ corresponding to $\gamma = \beta_2 \overline{\beta_1} / \operatorname{nrd}(I_\psi)$
Compute $P_f = [2^{g-f}]P_g$ and $Q_f = [2^{g-f}]Q_g$
Construct $(2^f, 2^f)$-isogeny $\Psi\colon E_1 \times E_2 \to \widetilde{E}_T \times E'$ with kernel
    $\langle([d_1]\varphi_1(P_f), \varphi_2 \circ \gamma(P_f)), ([d_1]\varphi_1(Q_f), \varphi_2 \circ \gamma(Q_f))\rangle$
Compute $\widetilde{E}_T$ via codomain of $\Psi$
return $(\widetilde{E}_T, \mathbf{r})$
```

The main idea of Algorithm 7 is to find equivalent ideals $I_1, I_2 \sim I_\psi$ of norms $d_1, d_2$ and isogenies $\varphi_1, \varphi_2$ of degrees $a_1, a_2$ in order to construct a 2-dimensional isogeny $\Psi$ of degree $a_1 d_1 + a_2 d_2 = 2^f$ that represents $\psi$. It uses

FullRepresentInteger from [23] to find $\gamma_1, \gamma_2 \in \operatorname{End} E_0$ with degrees $a_i(2^g - a_i)$. They are formally split into an isogeny $\varphi_i$ of degree $a_i$ and another isogeny of degree $2^g - a_i$. Then Kani's Lemma allows to compute $\varphi_1, \varphi_2$ and $\Psi$. A more detailed description is given in [5, Section 3.2].

## 4.3 Parameter Sizes and other Choices

The algorithms above do not specify all properties of the parameters. Therefore, we now discuss the necessary and some optional choices. For example, the hiding property sets requirements on the size of some parameters and we also propose some choices for implementing this protocol. In Appendix A we give an example how explicit numbers may look like.

The delay $t_{\mathrm{fd}}$ should be large, but it has to be polynomial in $\kappa$ (or $\log p$). On one hand the main idea of NITC schemes is that we can forcefully open a commitment (in polynomial time) with FDecom, if someone refuses to open it themselves. On the other hand generic algorithms to solve Problems 3.5 - 3.7 could be faster than FDecom and therefore violate hiding, if $t_{\mathrm{fd}}$ was superpolynomial. In particular, we need $t_{\mathrm{fd}} < d_s^{1/4}$ due to Assumptions 5.6 and 5.7 of Section 5.1 for quantum security and $t_{\mathrm{fd}} \gg t_{\mathrm{cv}}, t_{\mathrm{dv}}$.

### 4.3.1 Prime $p$, Starting Curve $E_0$ and Isogenies $\varphi_s$ and $\varphi_T$

The starting curve could be any supersingular elliptic curve $E_0$ with a known efficient representation of $\mathcal{O}_0$. For our protocol we choose $E_0$ to be the curve $E_0\colon y^2 = x^3 + x$ with $(p+1)^2$ points over $\mathbb{F}_{p^2}$ and $\mathcal{O}_0 = \langle 1, \mathrm{i}, \frac{\mathrm{i}+\mathrm{j}}{2}, \frac{1+\mathrm{k}}{2}\rangle_{\mathbb{Z}}$ for $p \equiv 3 \bmod 4$. In this case the endomorphisms $[\mathrm{i}]\colon (x,y) \mapsto (-x, \mathrm{i}y)$, $\phi\colon (x,y) \mapsto (x^p, y^p)$ (Frobenius map), $\theta$ and $\eta$ correspond to i, j, j + $\frac{1+\mathrm{k}}{2}$ and i, respectively[4]. This also allows for more efficient KLPT variants and higher dimensional IdealToIsogeny algorithms. In order to satisfy the hiding property, $p$ and $d_s$ would have a certain size. It has to be infeasible to precompute $\mathcal{O}_s$ for all possible $E_s$ or to find an isogeny from $E_0$ to $E_s$ in time less than $t_{\mathrm{fd}}$ in the online phase. Therefore, we choose $p \approx 2^{2\kappa}$, $2^\kappa \leq d_s \leq 2^{2\kappa}$ and $d_s \leq p$. In Section 5 we give a more detailed justification of these numbers.

Usually, we would want $d_s$ and $d_T$ to be smooth numbers both dividing $p+1$ in order to have fast evaluation of the corresponding isogenies. So $d_s$ should be smooth and divide $p+1$ (or $p^2-1$). However, evaluating $\varphi_T$ does not need to (in fact should not) be fast, since it is only evaluated by FDecom. Therefore, $d_T$ can contain larger prime factors and does not need to divide $p+1$ (or $p^2-1$). It is chosen such that computing an isogeny of degree $d_T$ takes at least time $t_{\mathrm{fd}}$ (and preferably not much longer). Note that we can choose $d_T$ as a large composite number with many prime factors to have more confidence in sequentiality. Since $P_T = \varphi_s(P_T')$ and $Q_T = \varphi_s(Q_T')$ have to generate $E_s[d_T]$, we need the degree $d_s$ of $\varphi_s$ to be coprime to $d_T$. Computing $K_T = \varphi_s(P_T' + [t]Q_T')$ can be slow if $t \approx d_T$, since computing $[t]Q_T'$ takes $O(\log t)$ operations. Thus, we just take $0 \leq t < d_s$.

For a supersingular curve with $(p+1)^2$ points over $\mathbb{F}_{p^2}$ we have $(p^e - (-1)^e)^2$ points over $\mathbb{F}_{p^{2e}}$ and the largest fully $\mathbb{F}_{p^{2e}}$-rational torsion group is the $(p^e - (-1)^e)$-torsion. If $d_T$ is large or contains prime factors that do not divide $p+$

[4] This is not a typo. We have $\eta = [\mathrm{i}]$.

1, this means that we need to go to extensions of $\mathbb{F}_{p^2}$ to find a basis for the $d_T$-torsion group of $E_0$ or $E_s$. Higher extensions and larger $p$ slow down the computations, therefore we want to minimize the degree of the extension and the size of $p$ to increase efficiency. The size of $p$ affects almost all computations, whereas the size of $e$ only influences computations related to the $K_T$ or $\varphi_T$. Both `Com` and `DecVrfy` have to compute $K_T$, but not $\varphi_T$. Thus, for longer delays it can be beneficial to make $d_T$ less smooth rather than making it larger. This can increase the delay without increasing the extension degree $e$, i.e. without slowing down `Com` and `DecVrfy`.

For an implementation we can choose a prime $p \equiv 3 \bmod 4$ such that $p+1$ contains a smooth factor $d_s = 2^\kappa$. This ensures that the first isogeny $\varphi_s \colon E_0 \to E_s$ can be evaluated efficiently. After choosing a prime, we find an extension degree $e$ such that $p^e - (-1)^e$ contains a suitable factor $d_T$ that is coprime to $d_s$. For example, we can use Assumption 5.4 to choose $d_T$ odd with prime factorization $d_T = \prod q_i^{e_i}$ such that $\sum e_i \sqrt{q_i} > t_{\mathrm{fd}}$. Hence, the size of $d_T$ and $e$ depend on the target delay $t_{\mathrm{fd}}$ and the smoothness of $d_T$. If $d_T$ is a prime, both $d_T$ and $e$ can potentially be small. But if $d_T$ is smooth it has to be large and requires a large extension degree $e$. The primes used in SIKE allow to choose $d_s$ (and $d_T$) this way. So there are already known primes with the right properties for different security levels. For $d_s \approx p$, the delay $t_{\mathrm{fd}}$ can be almost as large as $p^{1/4}$ instead of $p^{1/8}$. This could be a good trade-off for large delays.

**Remark 4.3.** *If we use the* 1*-dimensional* `IdealToIsogeny` *Algorithm 6, we need additional* $\mathbb{F}_{p^4}$*-rational torsion and have to use SQISign-friendly primes. In that case the maximal power of* 2 *that divides* $p+1$ *is often smaller than* $\sqrt{p}$*. This in turn requires using more involved methods (like [15, Algorithm 14]) for finding isogenies of degree* $d_s$*. Using higher dimensional isogenies as in Algorithm 7, however, only requires* $p = c2^k - 1$ *with* $c$ *as small as possible.*

### 4.3.2 Message Space $M$ and Function $F$

We choose $M$ to be a finite group $M = \mathbb{Z}/N\mathbb{Z}$ for an integer $N \in \mathbb{N}$. This gives us very efficient membership testing and group operations. The size of $N$ depends on the needed length of a message $m$ and the prime $p$. If $N$ is larger than $\lfloor p/12 \rfloor + 2$, then $F \colon J_{SS} \to M$ can not be surjective and therefore $u = m \ominus F(j_T)$ might leak information about the message $m$.

As mentioned before, computing $j_T$ from $F(j_T)$ has to be infeasible or at least slow. In order to satisfy hiding we choose the function $F$ to be $\lambda$-inverse resistant with $\lambda = \frac{3}{2}\kappa \approx \log p^{3/4}$. In addition, it has to be fast since `Com` and `DecVrfy` have to compute $F(j_T)$. An easy way to accomplish this is to take a function that is not injective. The larger the kernel of $F$, i.e. smaller $N$, the more information is lost. A simple projection $\mathbb{F}_{p^2} \supset J_{SS} \to \mathbb{F}_p$ onto one of the components or even their sum will leak information, since there is a subset of $j$-invariants that already are in $\mathbb{F}_p$. If we use a simple map like $(a,b) \mapsto b \bmod N$ or $(a,b) \mapsto a+b \bmod N$, we thus need to use $N \ll p$.

The right orders of equivalent ideals are only isomorphic and the corresponding shortcut isogenies have codomains with $j$-invariants that agree only up to Galois conjugacy. It is easier to choose $F$ oblivious to Galois conjugacy than to ensure equality of orders or $j$-invariants. For an implementation we can identify

$J_{SS} \subset \mathbb{F}_{p^2}$ with a subset of $\mathbb{F}_p[\mathrm{i}] \cong \mathbb{F}_{p^2}$ and choose

$$F\colon J_{SS} \to M = \mathbb{Z}/N\mathbb{Z}, \quad a + b\mathrm{i} \mapsto a + |b| \bmod N.$$

Since we choose $p \equiv 3 \bmod 4$ we have $(a + b\mathrm{i})^p = a - b\mathrm{i}$ and $F(j) = F(j^p)$. Also, $F$ should be $\frac{3}{2}\kappa$-inverse resistant, so we take $N \leq \lfloor p^{1/4}/12 \rfloor$. Then we can expect every element in $M$ to be the image of about $p^{3/4} \approx 2^{3\kappa/2}$ elements in $J_{SS}$. There is no direct way of finding the supersingular $j$-invariants. Hence, one would have to compute the preimage in $\mathbb{F}_{p^2}$ (about $12p^{7/4}$ elements) and check if they are $j$-invariants of supersingular elliptic curves. This is sufficiently inverse resistant in practice.

**Remark 4.4** (Publicly Verifiable)**.** *To make the scheme publicly verifiable, we can add an encryption* $\mathrm{Enc}_{j_T}(\pi_{\mathrm{dec}})$ *of* $\pi_{\mathrm{dec}}$ *to the commitment such that* $j_T$ *is the key for the decryption* $\mathrm{Dec}_{j_T}(\mathrm{Enc}_{j_T}(\pi_{\mathrm{dec}})) = \pi_{\mathrm{dec}}$*. This allows* `FDecom` *to provide the decommitment proof* $\pi_{\mathrm{dec}}$ *and everyone could use* `DecVrfy` *to verify the output of* `FDecom` *instead of computing it themselves.*

# 5 Security

We show that our protocol satisfies the Definition 2.1 of a NITC scheme by Katz et al. [31] and prove the three properties practicality, hiding and binding. These proofs are based on assumptions for the relative speed of some algorithms.

Our algorithms have the correct input and output arguments, and for all $\kappa$ and $m \in M$ every set of honestly generated $(\kappa, m, \mathbf{crs}, \mathbf{C}, \pi_{\mathrm{com}}, \pi_{\mathrm{dec}})$ satisfies verification $\mathtt{ComVrfy}(\mathbf{crs}, \mathbf{C}, \pi_{\mathrm{com}}) = \mathbf{accept} = \mathtt{DecVrfy}(\mathbf{crs}, \mathbf{C}, m, \pi_{\mathrm{dec}})$ and forced decommitment $\mathtt{FDecom}(\mathbf{crs}, \mathbf{C}) = m$. This makes it a NITC scheme.

## 5.1 Relative Running Times

Computing isogenies of prime degree $q$ can be done using Vélu's formulae in time $O(q)$, or the $\sqrt{\text{élu}}$ algorithm [6] in time $\sqrt{q}(\log q)^{2+o(1)}$ or $\widetilde{O}(\sqrt{q})$ for short. Here $\widetilde{O}$ may also contain additional logarithmic terms $\widetilde{O}(n) = O(n\,\mathrm{poly}(\log n))$. The crossover point for optimized algorithms is at $q \approx 100$, and we denote the time it takes to compute an isogeny of prime degree $q$ with $\mathrm{eval}_{\mathrm{prime}}(q)$. Remember that our timings are the number of operations rather than real-world times.

**Remark 5.1.** *If the kernel* $\ker\varphi$ *of an isogeny* $\varphi\colon E \to E'$ *or a point* $P \in E$ *in its domain is only* $\mathbb{F}_{p^{2e}}$*-rational, we need the same number of operations to find the codomain* $E/\ker\varphi$ *or to evaluate* $\varphi(P)$*, but they could be operations in* $\mathbb{F}_{p^{2e}}$ *instead of* $\mathbb{F}_{p^2}$*. The point* $K_T$ *of the commitment might only be defined over extension fields and we need to compute* $\varphi_s(K_T') = K_T$*. Therefore, we only considered the number of operations and do not distinguish between operations in* $\mathbb{F}_{p^2}$ *and more costly operations in extension fields* $\mathbb{F}_{p^{2e}}$*. The majority of operations of* `FDecom` *may be in extension fields, but for* `Com`*,* `ComVrfy` *and* `DecVrfy` *most operations can be done in* $\mathbb{F}_{p^2}$*. So our timings are rather conservative.*

**Lemma 5.2.** *There is a (small) constant* $c_p$ *such that evaluating an isogeny of prime degree* $q$ *takes time* $\mathrm{eval}_{\mathrm{prime}}(q) \leq c_p q$*.*

Now let us look at an isogeny $\varphi$ with a kernel that is generated by a point $K_0'$ of order $q^k$. We can decompose $\varphi = \varphi_k \circ \cdots \circ \varphi_1$ into isogenies $\varphi_i$ of degree $q$. In each step we compute the points $K_i = [q^{k-i}]K_{i-1}'$ generating the kernel of $\varphi_i$ and $K_i' = \varphi_i(K_{i-1}')$ generating the kernel of $\varphi_i' = \varphi_k \circ \cdots \circ \varphi_{i+1}$. So every step takes time $\mathrm{eval}_{\mathrm{prime}}(q)$ plus the time it takes to compute the point multiplication. Generalizing this to isogenies of arbitrary composite degree gives us bounds for the time $\mathrm{eval}(d)$ it takes to compute an isogeny of degree $d$. If we ignore the multiplications for the lower bound we get the following lemma.

**Lemma 5.3.** *Let $d = \prod_{i=1}^r q_i^{e_i}$ be the prime factorization of the degree $d$. There is a (small) constant $c_c \geq 1$ such that the time* $\mathrm{eval}(d)$ *it takes to evaluate an isogeny of degree $d$ is bounded by*

$$\sum_{i=1}^r e_i \, \mathrm{eval}_{\mathrm{prime}}(q_i) \leq \mathrm{eval}(d) \leq c_c \sum_{i=1}^r e_i \, \mathrm{eval}_{\mathrm{prime}}(q_i).$$

This allows us to choose $d_T$ such that $\mathrm{eval}(d_T) \geq t_{\mathrm{fd}}$. If we assume that Vélu's formulae and the $\sqrt{\text{é}}\text{lu}$ algorithm are close to optimal in computing prime degree isogenies we can use the following assumption to choose values for $d_T$.

**Assumption 5.4.** *Let $d = \prod_{i=1}^r q_i^{e_i}$ be the prime factorization of the degree $d$. We assume that $\sqrt{q}_i \leq \mathrm{eval}_{\mathrm{prime}}(q_i)$ and hence $\sum_{i=1}^r e_i \sqrt{q_i} \leq \mathrm{eval}(d)$.*

Combining these results we get an upper bound for the computation time of isogenies of smooth degree.

**Lemma 5.5.** *An isogeny of degree $d$ with prime factorization $d = \prod_{i=1}^r q_i^{e_i}$ and $q_i < B$ ($B$-smooth) can be evaluated in time $O(\frac{B}{\log B} \log d)$.*

*Proof.* We use Lemmas 5.3 and 5.2 to write

$$\mathrm{eval}(d) \leq c_c \sum_{i=1}^r e_i \, \mathrm{eval}_{\mathrm{prime}}(q_i) \leq c_c c_p \sum_{i=1}^r e_i q_i.$$

Since $q_i < B$ for all $1 \leq i \leq r$, we get $q_i \leq \log q_i \frac{B}{\log B}$ and

$$\mathrm{eval}(d) \leq c_c c_p \sum_{i=1}^r e_i \log q_i \frac{B}{\log B} = c_c c_p \frac{B}{\log B} \log d. \qquad \square$$

According to Eisenträger et al. [27] the fastest (currently known) algorithms for solving the (equivalent) general Isogeny Path Problem, general Endomorphism Ring Problem or general Maximal Order Problem (cf. Section 3.2) over $\mathbb{F}_{p^2}$ take time $\widetilde{O}(p^{1/2})$ for classical computations and $\widetilde{O}(p^{1/4})$ with a quantum computer. Since $E_0$ and $E_s$ are known to be connected by a $d_s$-isogeny there is also a meet-in-the-middle or claw-finding attack in classical time $\widetilde{O}(d_s^{1/2})$ and $\widetilde{O}(d_s^{1/4})$ when applying Grover's Algorithm [29].

**Assumption 5.6** (General Isogeny Assumption)**.** *We assume that the fastest algorithms to solve the general Isogeny Path Problem, the general Endomorphism Ring Problem or the general Maximal Order Problem over $\mathbb{F}_{p^2}$ need at least $p^{1/2}$ or $p^{1/4}$ operations for classical or quantum algorithms, respectively.*

**Assumption 5.7** (Special Isogeny Assumption)**.** *We assume that the fastest algorithms to find an isogeny between two $d$-isogenous curves over $\mathbb{F}_{p^2}$ with $d < p$ take at least $d^{1/2}$ or $d^{1/4}$ operations for classical or quantum algorithms, respectively.*

With these assumptions we can prove that computing the codomain of an isogeny can be made almost arbitrarily slow.

**Theorem 5.8.** *Let $E$ be a supersingular elliptic curve over $\mathbb{F}_{p^2}$ with unknown $\mathcal{O} \cong \operatorname{End} E$, but $d'$-isogenous to a curve $E_0$ with known endomorphism ring. Let further $K$ be a point on $E$ of order $d$, such that computing the corresponding isogeny takes at least time $t$, according to Lemma 5.3. Then for $t < \min\{d'^{1/4}, p^{1/4}\}$ and under Assumptions 5.6 and 5.7, computing $E_K \cong E/\langle K\rangle$ takes at least time $t$.*

*Proof.* The isogeny $\varphi\colon E \to E_K$ with kernel $\langle K\rangle$ has degree $d$. Efficiently calculating a shortcut isogeny $\widetilde{\varphi}\colon E \to E_K$ or an efficient higher dimensional representation requires knowledge of $\mathcal{O} \cong \operatorname{End} E$. Finding the endomorphism ring $\operatorname{End} E$ or the order $\mathcal{O} \cong \operatorname{End} E$ without an isogeny $\varphi'\colon E_0 \to E$, or finding an isogeny $\widetilde{\varphi}$ without $\mathcal{O} \cong \operatorname{End} E$ are hard problems. By Assumption 5.6 finding $\operatorname{End} E$ or $\mathcal{O}$ takes time at least $p^{1/4} > t$. Finding an isogeny $\varphi'$ needs at least time $d'^{1/4} > t$ by Assumption 5.7 if $d' < p$ or $p^{1/4} > t$ by Assumption 5.6 if $d' \geq p$. Therefore, computing $E_K \cong E/\langle K\rangle$ takes at least time $t$. □

The algorithm `FDecom` only has **crs** and $\mathbf{C} = (E_s, K_T, u)$ as input. In order to compute $m = u \oplus F(j_T)$ it has to calculate the $j$-invariant $j_T$ of the secret curve $E_T \cong E_s/\langle K_T\rangle$. Theorem 5.8 gives us the following corollary:

**Corollary 5.9.** *For $t_{\text{fd}} < d_s^{1/4}$ and under the Assumptions 5.6 and 5.7, the forced decommitment* `FDecom` *takes at least time $t_{\text{fd}}$.*

Note that the restriction $t_{\text{fd}} < d_s^{1/4}$ is based on the quantum timings in Assumptions 5.6 and 5.7. For classical algorithms $t_{\text{fd}} < d_s^{1/2}$ would be sufficient, but since our protocol should be quantum secure we chose the more general bound including quantum algorithms.

## 5.2 Hiding and Binding

For hiding we use the same (non-malleability) Definition 2.4 as Katz et al. [31]. First we show why we need an adapted security game. In Definition 2.3 the adversary $\mathcal{A}$ sends two messages $m_0, m_1$ and receives the commitment $\mathbf{C}_b = (E_s, K_T, u_b)$ corresponding to message $m_b$ for a uniform $b \in \{0, 1\}$. It is allowed to query an oracle for `FDecom`$(\cdot)$ except for `FDecom`$(\mathbf{crs}, \mathbf{C}_b)$.

**Lemma 5.10.** *An adversary $\mathcal{A}$ can break hiding with the original security game from Definition 2.3.*

*Proof.* Since $m_{1-b} \ominus m_b \oplus u_b = u_{1-b}$, querying `FDecom`$(\mathbf{crs}, (E_s, K_T, u_\pm))$ with $u_+ = (m_0 \ominus m_1) \oplus u_b$ and $u_- = \ominus(m_0 \ominus m_1) \oplus u_b$ gives $m_{1-b}$ and a random message $m'$. For $|M| = 2$ we have $u_+ = u_-$ and get $m_{1-b}$. For $|M| > 2$ however, we can assume $m_0 \neq m' \neq m_1$. This allows $\mathcal{A}$ to output the correct $b' = b$ with high probability.

Even worse, if we replace $K_T$ by any other point $K'$ such that $\langle K' \rangle = \langle K_T \rangle$, e.g. $K' = [\ell]K_T$ for $\ell$ coprime to $d_T$, or apply an isomorphism such that $E'/\langle K' \rangle \cong E_T \cong E_s/\langle K_T \rangle$ then $\texttt{FDecom}(\mathbf{crs}, (E', K', u_b))$ will return $m_b$. □

Thus, it is reasonable to disallow queries of the form $\texttt{FDecom}(\mathbf{crs}, (E', K', \cdot))$ for $E'/\langle K' \rangle \cong E_T \cong E_s/\langle K_T \rangle$. Since $F$ is oblivious to Galois conjugacy, we also disallow queries with $E'/\langle K' \rangle \cong E_T^p$ and use the adapted security game in Definition 5.11.

**Definition 5.11** (IND-CCA adapted)**.** *For a NITC scheme* $\texttt{TC}$ *and an algorithm* $\mathcal{A}$*, define the game* $\texttt{IND-CCA}_{\texttt{TC}}^{\mathcal{A}}$ *as follows:*

1. *Compute* $\mathbf{crs} \leftarrow \texttt{PGen}(1^\kappa)$*.*

2. *Run* $\mathcal{A}(\mathbf{crs})$ *in a preprocessing phase with access to* $\texttt{FDecom}(\mathbf{crs}, \mathbf{C})$ *for valid commitments* $\mathbf{C} = (E, K, u)$ *with* $\operatorname{ord} K \mid d_T$*.*

3. *When* $\mathcal{A}$ *outputs* $(m_0, m_1)$*, choose a uniform bit* $b \leftarrow \{0, 1\}$ *and then compute* $(\mathbf{C}_b, \pi_{\text{com}}, \pi_{\text{dec}}) \leftarrow \texttt{Com}(\mathbf{crs}, m_b)$*. Give* $(\mathbf{C}_b, \pi_{\text{com}})$ *to* $\mathcal{A}$*, who continues to have access to* $\texttt{FDecom}(\mathbf{crs}, \mathbf{C})$ *for valid commitments* $\mathbf{C} = (E, K, u)$ *with* $\operatorname{ord} K \mid d_T$ *except that it may not query the oracle on* $(E', K', \cdot)$ *for* $E'/\langle K' \rangle$ *isomorphic to* $E_s/\langle K_T \rangle$ *or its Galois conjugate where* $\mathbf{C}_b = (E_s, K_T, u_b)$*.*

4. *When* $\mathcal{A}$ *outputs a bit* $b'$*, it wins iff* $b' = b$*.*

This is still in the spirit of the original definition, since it prohibits the “decryption” of the commitment in question. In our case the security arises from the secret isogeny $\varphi_s \colon E_0 \to E_s$ and the delay isogeny $\varphi_T \colon E_s \to E_T$ with kernel $\langle K_T \rangle$, and the “key” is $F(j_T)$ for $j_T = j(E_T)$. Such queries would enable $\mathcal{A}$ to find $F(j_T)$ and would hence basically allow to query $\texttt{FDecom}(\mathbf{crs}, (E_s, K_T, u_b))$ by proxy, which is forbidden in the original definition.

**Remark 5.12.** *The restriction of oracle queries to valid commitments* $\mathbf{C} = (E, K, u)$ *with* $\operatorname{ord} K \mid d_T$ *ensures that the oracle can be simulated in polynomial time. This is a reasonable restriction, as* $\texttt{FDecom}$ *in the original definition can output* **invalid** *for malformed inputs. We can even check* $\operatorname{ord} K \mid d_T$ *in* $\texttt{ComVrfy}$ *as mentioned in Remark 4.2.*

**Theorem 5.13.** *For a* $\frac{3}{2}\kappa$*-inverse resistant function* $F$ *and under the Assumptions 5.6 and 5.7, SIGNITC is* $(t_p, t_o, \varepsilon)$*-CCA-secure (satisfies hiding) with security game from Definition 5.11 for* $t_p \ll 2^\kappa$ *polynomial in* $\kappa$*,* $t_o < t_{\text{fd}}$ *and* $\varepsilon = 2^{-\kappa}$*.*

*Proof.* The precomputation phase can only provide a negligible advantage for an adversary $\mathcal{A}$. The computation of $\texttt{Com}(\mathbf{crs}, m)$ includes choosing random $K_s = P_s + [s]Q_s \in E_0[d_s]$ and $K_T = P_T + [t]Q_T \in E_s[d_T]$ with $s, t \in [0, d_s)$. Since $2^\kappa \leq d_s$, it is infeasible to precompute (and store) a significant subset of all possibilities in time $t_p \ll 2^\kappa$ polynomial in $\kappa$.

In the online phase $\mathcal{A}$ sends two messages $m_0, m_1$ and receives the output $(E_s, K_T, u_b)$ of $\texttt{Com}(\mathbf{crs}, m_b)$ for a uniform $b \in \{0, 1\}$. The adversary $\mathcal{A}$ knows that $F(j_T)$ is equal to $F_0 = \ominus u_b \oplus m_0$ or $F_1 = \ominus u_b \oplus m_1$. Since $F$ is a $\frac{3}{2}\kappa$-inverse resistant function, there are at least $2^{3\kappa/2}$ $j$-invariants $j$ such that $F(j) = F_i$ for each $i \in \{0,1\}$ and none of them is more likely than the other. To verify one of them, $\mathcal{A}$ would have to compute $E_s/\langle K_T\rangle$, but this is equivalent to computing $\texttt{FDecom}(\mathbf{crs}, (E_s, K_T, u_b))$. Under Assumptions 5.6 and 5.7 we get that it can not be done in time $t_o$ less than $t_{\mathrm{fd}}$ by Corollary 5.9.

We can choose the smallest prime $\ell \mid d_T$ and find $E, K$ such that $j = j(E/\langle K\rangle)$ is one of the $\ell+1$ neighbors of $j_T$ in the $\ell$-isogeny graph. Then we can compute the $\ell+1$ neighbors $j_k$ of $j$ and check if $F(j_k)$ matches $F_0$ or $F_1$. If we have only one match, then this gives $F(j_T)$. If we have more matches, then we have to try again with a different $j$. To increase our confidence in a candidate, we can repeat this for more or all $\ell+1$ neighbors of $j_T$, since $F(j_T)$ has to be a match for a neighbor for all of them. The easiest way to find such $E$ and $K$ is to take $E = E_s$ and $K = [\ell]K_T$. If we need more neighbors, we can also compute the isogeny $\varphi_\ell\colon E_s \to E$ of degree $\ell$ with kernel $\langle[\frac{d_T}{\ell}]K_T\rangle$ and take $E = E_s/\langle[\frac{d_T}{\ell}]K_T\rangle$. For the point $K$ we take one of the points in $E$ such that $[\ell]K = \varphi_\ell(K_T)$. The problem with this approach is that computing $E$, $K$, $E/\langle K\rangle$ and $\ell+1$ isogenies of degree $\ell$ is slower than computing $E_s/\langle K_T\rangle$. As discussed above, we assume that this can not be done in time $t_o$ less than $t_{\mathrm{fd}}$.

If we query the oracle on $(E, K, 0)$ instead of computing $E/\langle K\rangle$, then we only get $F(j) = 0 \oplus F(j)$ instead of $j = j(E/\langle K\rangle)$. Since $F$ is an $\frac{3}{2}\kappa$-inverse resistant function, there are at least $2^{3\kappa/2}$ indistinguishable candidates for each $j$. In this case the best approach is to find $\ell+1$ pairs $(E_k, K_k)$ such that $j_k = j(E_k/\langle K_k\rangle)$ are the $\ell+1$ neighbors of $j_T$ in the $\ell$-isogeny graph. Then we query the oracle on all of them to get $F(j_k)$ for $1 \le k \le \ell+1$. Now we have to match them with the neighbors of the (at least) $2^{3\kappa/2}$ candidates for $j_T$ from each $F_0$ and $F_1$. To have confidence in a candidate, its neighbors have to match with several or all $F(j_k)$. If $F_0 \neq F_1$, then their preimages are disjoint and there are at least $2 \cdot 2^{3\kappa/2}$ candidates for $j_T$. We can only spend less than $t_o < t_{\mathrm{fd}} < d_s^{1/4} \le 2^{\kappa/2}$ operations on these comparisons. Therefore, the probability to find $j_T$ is less than $t_o 2^{-3\kappa/2} < 2^{-\kappa}$.

When we replace $E_s$ and $K_T$ by a curve $E'$ and point $K'$ such that $E'/\langle K'\rangle$ is unrelated to $E_s/\langle K_T\rangle$, the query $\texttt{FDecom}(\mathbf{crs}, (E', K', u_b))$ gives completely unrelated results. In conclusion, in the online phase the advantage over guessing is less than $2^{-\kappa}$ under Assumptions 5.6 and 5.7. □

The new security game for binding in Definition 5.14 only restricts oracle access to valid commitments in order to allow efficient simulations as mentioned in Remark 5.12.

**Definition 5.14** (BND-CCA adapted)**.** *For a NITC scheme* TC *and an algorithm $\mathcal{A}$, define the game* $\texttt{BND-CCA}^{\mathcal{A}}_{\texttt{TC}}$ *as follows:*

1. *Compute* $\mathbf{crs} \leftarrow \texttt{PGen}(1^\kappa)$*.*
2. *Run* $\mathcal{A}(\mathbf{crs})$ *with access to* $\texttt{FDecom}(\mathbf{crs}, \mathbf{C})$ *for valid commitments* $\mathbf{C} = (E, K, u)$ *with* $\operatorname{ord} K \mid d_T$*.*
3. *$\mathcal{A}$ outputs* $(m, \mathbf{C}, \pi_{\mathrm{com}}, \pi_{\mathrm{dec}}, m', \pi'_{\mathrm{dec}})$ *and wins iff* $\texttt{ComVrfy}(\mathbf{crs}, \mathbf{C}, \pi_{\mathrm{com}}) =$ **accept** *and either:*
   - *$m \neq m'$, yet* $\texttt{DecVrfy}(\mathbf{crs}, \mathbf{C}, m, \pi_{\mathrm{dec}})$ *and* $\texttt{DecVrfy}(\mathbf{crs}, \mathbf{C}, m', \pi'_{\mathrm{dec}})$ *both output* **accept***;*

- $\texttt{DecVrfy}(\mathbf{crs}, \mathbf{C}, m, \pi_{\mathrm{dec}}) = \mathbf{accept}$ *but* $\texttt{FDecom}(\mathbf{crs}, \mathbf{C}) \neq m$.

The proof for binding works with the original Definition 2.6 and the new security game from Definition 5.14. With our protocol we even achieve perfect binding.

**Theorem 5.15.** *SIGNITC is* $(\infty, 0)$*-BND-CCA-secure (satisfies binding) with security game from Definition 5.14.*

*Proof.* If the commitment $\mathbf{C}$ is accepted by ComVrfy, then it contains an elliptic curve $E_s$, a point $K_T$ on $E_s$ and an element $u$ of an additive group $M$. Since DecVrfy verifies $u \oplus F(j_T) = m$ for $j_T = j(E_s/\langle K_T \rangle)$, we have that acceptance of both $(\mathbf{crs}, \mathbf{C}, m, \pi_{\mathrm{dec}})$ and $(\mathbf{crs}, \mathbf{C}, m', \pi'_{\mathrm{dec}})$ by DecVrfy implies $m \ominus F(j_T) = u = m' \ominus F(j_T)$ and hence $m = m'$. The speedup does not change this, because it can only change $j_T$ by Galois conjugacy and $F$ is oblivious to that. Similarly, if DecVrfy accepts $(\mathbf{crs}, \mathbf{C}, m, \pi_{\mathrm{dec}})$ then $u = m \ominus F(j_T)$. FDecom computes $F(j_T)$ from $E_s$ and $K_T$ and thus outputs the correct message $m = u \oplus F(j_T)$. □

## 5.3 Practicality

We show that Com, ComVrfy and DecVrfy can be computed efficiently and that we achieve a perfectly practical NITC scheme. We chose $p \approx 2^{2\kappa}$, $2^{\kappa} \leq d_s \leq 2^{2\kappa}$, $d_s \leq p$ and $t_{\mathrm{fd}} < d_s^{1/4}$ to get $\kappa$ bits of classical and $\kappa/2$ bits of quantum security for the precomputation phase in hiding. In this subsection "efficiently" means an expected running time of at most $\mathrm{poly}(\log p)$ operations for probabilistic algorithms.

**Lemma 5.16.** *The commitment* Com *takes time* $t_{\mathrm{com}} \in \mathrm{poly}(\log p)$*.*

*Proof.* The number of operations on $E_0$ for computing $K_s = P_s + [s]Q_s$ and $K'_T = P'_T + [t]Q'_T$ is linear in $\log d_s$ since $0 \leq s, t < d_s$. By Lemma 5.5 we can find $E_s \cong E_0/\langle K_s \rangle$ and $K_T = \varphi_s(K'_T)$ via Vélu's formulae in time $O(\log d_s)$ since $d_s$ is a power of 2 and hence smooth. We adapted Algorithm 23 from [15] to compute $I_s$ and $I'_T$ from $K_s$ and $K'_T$ using each one inversion and a few additions and multiplications modulo $d_s$ or $d_T$, respectively. The IdealToIsogeny algorithms are in $\mathrm{poly}(\log p)$. Finding the isogenies can be done efficiently with the algorithms from SQIsign [15] or SQIsign2D-West [5] for Algorithms 6 or 7, and the evaluation of the resulting chains of 2- or $(2, 2)$-isogenies is also efficient. Finally, we have to compute $\widetilde{j}_T = j(\widetilde{E}_T)$ and $u = m \ominus F(\widetilde{j}_T)$. Since we chose $F$ and the group operation in $M$ to be efficiently computable and $d_s \leq p$, we get that the algorithm takes time $t_{\mathrm{com}} \in \mathrm{poly}(\log p)$. □

**Lemma 5.17.** *The maximal number of operations* $t_{\mathrm{cv}}$ *for algorithm* ComVrfy *is a small constant.*

*Proof.* The algorithm has to complete three tasks. First it has to check if $E_s$ is an elliptic curve. To do that, it suffices to check that the discriminant is non-zero. For curves in short Weierstraß form $E\colon y^2 = x^3 + Ax + B$ this is just $4A^3 \neq -27B^2$. To check if $K_T$ is a point on $E_s$ it can simply compute if $K_T$ satisfies the curve equation. Finally, membership testing for $u \in M$ is efficient by definition of $M$. For $M = \mathbb{Z}/N\mathbb{Z}$ this means checking if $u$ is an integer (and if $0 \leq u < N$). So all this can be done in very few operations and their number is independent of the size of $d_s$, $d_T$, $p$ and $\kappa$. □

**Remark 5.18.** *If we also check* $\operatorname{ord} K_T \mid d_T$ *as suggested in Remark 4.2, we need to compute* $[d_T]K_T$*. This takes* $O(\log d_T)$ *operations, but for sufficiently non-smooth* $d_T = \prod q_i^{e_i}$ *we can assume that* $\log d_T \ll \sum e_i\sqrt{q_i}$*, i.e. scalar multiplication by* $d_T$ *is faster than evaluating an isogeny of degree* $d_T$*. In this case* `ComVrfy` *is still faster than* `FDecom`*.*

**Lemma 5.19.** *The decommitment verification algorithm* `DecVrfy` *takes time* $t_{\mathrm{dv}} \in \operatorname{poly}(\log p)$*.*

*Proof.* The decommitment verification has the same steps as the commitment. There are only three differences: Firstly, it gets $s, t$ from $\pi_{\mathrm{dec}}$ instead of choosing them and hence does not need to try again for bad choices of $t$. Secondly, it has to compare the $E_s$ and $K_T$ it computes to the ones in the commitment and $m$ to the decommitment. And thirdly, it computes `IdealToIsogeny` as deterministic algorithm using $\mathbf{r}$ from $\pi_{\mathrm{dec}}$. This makes its version of `IdealToIsogeny` faster. Since these differences are computationally insignificant we get that the algorithm also takes time $t_{\mathrm{dv}} \in \operatorname{poly}(\log p)$. □

Note that the running times of `Com`, `ComVrfy` and `DecVrfy` are not dominated by $d_T$. Even for low security levels like $\kappa = 128$ we get that $\log p \ll p^{1/8} \lesssim d_s^{1/4}$. Since $t_{\mathrm{fd}}$ can be almost as large as $d_s^{1/4}$, the previous Lemmas 5.16, 5.17 and 5.19 show that we can choose $t_{\mathrm{fd}}$ such that $t_{\mathrm{com}}, t_{\mathrm{cv}}, t_{\mathrm{dv}} \ll t_{\mathrm{fd}}$. This gives us the following theorem:

**Theorem 5.20.** *SIGNITC is perfectly practical under Assumptions 5.6 and 5.7.*

Assumption 5.4 allows us to choose $d_T = \prod_{i=1}^{r} q_i^{e_i}$ more explicitly such that $t_{\mathrm{com}}, t_{\mathrm{cv}}, t_{\mathrm{dv}} \ll \sum_{i=1}^{r} e_i\sqrt{q_i}$ (and $\log d_T \ll \sum e_i\sqrt{q_i}$ if we want to check $\operatorname{ord} K_T \mid d_T$ in `ComVrfy`).

# Conclusion

We showed that SIGNITC is a perfectly practical NITC that satisfies hiding and perfect binding. It is the first NITC without repeated squaring or black box algorithms, it needs no trusted setup and all subroutines have already been implemented for other cryptosystems. Since it uses only isogeny-based cryptography, it is presumably quantum secure. Since repeated squaring might not be a good candidate for creating a delay anymore, this could also be an interesting starting point for isogeny-based delay in other settings.

Finally, we list some open topics for further research. The most obvious one is to implement this protocol to get some benchmarks for (relative) real-world timings and to choose some specific parameters. Another open question is how this can be optimized. A recent paper [12] introduces new algorithms that can potentially improve the precomputations in `PGen`. We only looked at the number of operations regardless of the field extensions. Since working in higher extension fields can significantly slow down the computations, it might be beneficial to represent our isogenies other than with one point in a (possibly large) extension field. Other optimizations might include different ways to find a corresponding ideal given an isogeny, and improvements to KLPT like [36].

Choosing the best `IdealToIsogeny` algorithm for SIGNITC is another open question. KLPT and similar methods used in SQISign give ideals of norm $d \geq p^3$

so the $d$-torsion is not $\mathbb{F}_{p^4}$-rational. Computing an isogeny corresponding to such an ideal is the bottleneck in SQISign and there are already several approaches to improve this step. SQISign computes them in blocks and AprèsSQI [17] tries to reduce the number of blocks. In [28] they try to reduce the needed extension degree and give some general speedups. Other approaches [5, 19, 26, 33, 34, 35] use higher dimensional isogenies to avoid (higher) extension fields or to lift smoothness requirements on the norm $d$ of the equivalent ideal. We only need to know (the $j$-invariant of) the codomain and do not need to evaluate the isogeny on any points. So it would be interesting to see if these approaches can improve Algorithms 6 and 7.

# Acknowledgments

The author would like to thank Antonio Sanso for pointing out the concept of NITC schemes, Valerie Gilchrist, Lorenz Panny and anonymous reviewers for their comments on previous versions of this article, Max Duparc for interesting discussions on pushforwards and higher dimensional isogenies, and Jens Zumbrägel for his support during the work on this article.

# A Example Parameters

This is an example of how realistic parameters might look like. These are all based on estimates and assumptions. An implementation is needed to choose definitive parameters. We give three sets of parameters. One for the 1-dimensional `IdealToIsogeny` Algorithm 6 using a SQISign-friendly prime and two for the 2-dimensional `IdealToIsogeny` Algorithm 7 with primes for higher dimensional versions of SQISign.

## A.1 1-dimensional `IdealToIsogeny`

For a security of NIST level I with $\kappa = 128$ we can take the SQIsign-friendly prime $p = p^I_{1973}$ with

$p^I_{1973} =$ `0x34e29e286b95d98c33a6a86587407437252c9e49355147ffffffffffffffff`

and $\log_2 p^I_{1973} \approx 251.9$ from the specifications of SQIsign [15]. The $\mathbb{F}_{p^4}$-rational torsion is

$$\begin{aligned} p^2 - 1 = 2^{76} &\cdot 3^{36} \cdot 7^4 \cdot 11 \cdot 13 \cdot 23^2 \cdot 37 \cdot 59^2 \cdot 89 \cdot 97 \cdot 101^2 \cdot 107 \cdot 109^2 \cdot 131 \cdot 137 \\ &\cdot 197^2 \cdot 223 \cdot 239 \cdot 383 \cdot 389 \cdot 491^2 \cdot 499 \cdot 607 \cdot 743^2 \cdot 1033 \cdot 1049 \cdot 1193 \\ &\cdot 1913^2 \cdot 1973 \cdot 32587069 \cdot 275446333 \cdot 1031359276391767 \end{aligned}$$

and we choose $d_s = 2^{150}$. As mentioned in Remark 4.3, we have to use a method similar to [15, Algorithm 14] to produce isogenies of degree $d_s$ as the maximal power of 2 that divides $p+1$ is $2^{75}$. For Algorithm 6 we use the same additional $\mathbb{F}_{p^4}$-torsion

$$\begin{aligned} &3^{36} \cdot 7^4 \cdot 11 \cdot 13 \cdot 23^2 \cdot 37 \cdot 59^2 \cdot 89 \cdot 97 \cdot 101^2 \cdot 107 \cdot 109^2 \cdot 131 \cdot 137 \cdot 197^2 \cdot 223 \\ &\cdot 239 \cdot 383 \cdot 389 \cdot 491^2 \cdot 499 \cdot 607 \cdot 743^2 \cdot 1033 \cdot 1049 \cdot 1193 \cdot 1913^2 \cdot 1973 \end{aligned}$$

for translating as in SQIsign. For the group $M = \mathbb{Z}/N\mathbb{Z}$ we need $N$ smaller than 1036363420827959282, e.g. $N = 2^{59}$. The delay has to be $t_{\text{fd}} \leq 194368031998$. Using `Sage` for computing isogenies we can choose

$$\begin{aligned} d_T = 7^4 &\cdot 11 \cdot 13 \cdot 37 \cdot 89 \cdot 97 \cdot 107 \cdot 131 \cdot 137 \cdot 223 \cdot 239 \cdot 383 \cdot 389 \cdot 499 \cdot 607 \\ &\cdot 1033 \cdot 1049 \cdot 1193 \cdot 1973 \cdot 32587069 \cdot 275446333 \end{aligned}$$

for a delay of roughly 1 minute and we estimate that $d_T = 1031359276391767$ will cause a delay of roughly 1 day. Both divide $p - 1$ so we only need extension degree $e = 2$ and can work in $\mathbb{F}_{p^4}$ or even in $\mathbb{F}_{p^2}$ using quadratic twists.

### A.2 2-dimensional `IdealToIsogeny`

For a security of NIST level I with $\kappa = 128$ we have several options. We could choose the prime $p = 2^{216}3^{137} - 1$ from the SIKEp434 parameter set [30], but for 2-dimensional `IdealToIsogeny` algorithms primes of the form $p = c2^k - 1$ with $c$ as small as possible are preferable. We can choose $p = 79 \cdot 2^{247} - 1$ as in [35] or we take the prime $p = 5 \cdot 2^{248} - 1$ from SQIsign2D-West [5].

For $p = 5 \cdot 2^{248} - 1$ we choose $d_s = 2^{248}$ and $N \leq 574673255585861476$ for the group $M = \mathbb{Z}/N\mathbb{Z}$, e.g. $N = 2^{58}$. The delay has to be $t_{\text{fd}} < 2^{62}$ and we can choose $d_T = 5 \cdot 7 \cdot 3631 \cdot 2857849$ for $e = 3$.

For $p = 79 \cdot 2^{247} - 1$ we choose $d_s = 2^{247}$ and $N \leq 963446845306433641$ for the group $M = \mathbb{Z}/N\mathbb{Z}$, e.g. $N = 2^{59}$. The delay has to be $t_{\text{fd}} \leq 3877950241171266237$ and we can choose $d_T = 7 \cdot 13 \cdot 19 \cdot 79 \cdot 21313$ for $e = 3$.

Estimating the actual delay is difficult due to the field extensions. Finding primes $p$ such that $p + 1$ contains a large power of 2 and $p^2 - 1$ contains some primes of size at most $2^{60}$ would allow for different delays within $\mathbb{F}_{p^4}$. This is related to the search for SQISign-friendly primes, e.g. [1, 10, 15, 18, 22].

# References

[1] Knud Ahrens. Sieving for large twin smooth integers using single solutions to prouhet-tarry-escott. Cryptology ePrint Archive, Paper 2023/219, 2023. URL `https://eprint.iacr.org/2023/219`.

[2] Knud Ahrens and Jens Zumbrägel. DEFEND: Towards verifiable delay functions from endomorphism rings. Cryptology ePrint Archive, Paper 2023/1537, 2023. URL `https://eprint.iacr.org/2023/1537`.

[3] Miguel Ambrona, Marc Beunardeau, and Raphaël R. Toledo. Timed commitments revisited. Cryptology ePrint Archive, Paper 2023/977, 2023. URL `https://eprint.iacr.org/2023/977`.

[4] Andrea Basso, Giulio Codogni, Deirdre Connolly, Luca De Feo, Tako Boris Fouotsa, Guido Maria Lido, Travis Morrison, Lorenz Panny, Sikhar Patranabis, and Benjamin Wesolowski. Supersingular curves you can trust. In Carmit Hazay and Martijn Stam, editors, *Advances in Cryptology – EUROCRYPT 2023*, pages 405–437, Cham, 2023. Springer Nature Switzerland.

[5] Andrea Basso, Pierrick Dartois, Luca De Feo, Antonin Leroux, Luciano Maino, Giacomo Pope, Damien Robert, and Benjamin Wesolowski. Sqisign2d–west. In Kai-Min Chung and Yu Sasaki, editors, *Advances in Cryptology – ASIACRYPT 2024*, pages 339–370, Singapore, 2025. Springer Nature Singapore. ISBN 978-981-96-0891-1. doi: 10.1007/978-981-96-0891-1_11.

[6] Daniel J. Bernstein, Luca De Feo, Antonin Leroux, and Benjamin Smith. Faster computation of isogenies of large prime degree. In Steven D. Galbraith, editor, *Proceedings of the Fourteenth Algorithmic Number Theory Symposium*, pages 39–55, Berkeley, 2020. Mathematical Sciences Publishers. doi: 10.2140/obs.2020.4.39.

[7] Alex Biryukov, Ben Fisch, Gottfried Herold, Dmitry Khovratovich, Gaëtan Leurent, María Naya-Plasencia, and Benjamin Wesolowski. Cryptanalysis of algebraic verifiable delay functions. In Leonid Reyzin and Douglas Stebila, editors, *Advances in Cryptology – CRYPTO 2024*, pages 457–490, Cham, 2024. Springer Nature Switzerland. ISBN 978-3-031-68382-4. doi: 10.1007/978-3-031-68382-4_14.

[8] Dan Boneh and Moni Naor. Timed commitments. In Mihir Bellare, editor, *Advances in Cryptology – CRYPTO 2000*, pages 236–254, Berlin, Heidelberg, 2000. Springer Berlin Heidelberg. doi: 10.1007/3-540-44598-6_15.

[9] Dan Boneh, Joseph Bonneau, Benedikt Bünz, and Ben Fisch. Verifiable delay functions. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology – CRYPTO 2018*, pages 757–788, Cham, 2018. Springer International Publishing. doi: 10.1007/978-3-319-96884-1_25.

[10] Giacomo Bruno, Maria Corte-Real Santos, Craig Costello, Jonathan Komada Eriksen, Michael Meyer, Michael Naehrig, and Bruno Sterner. Cryptographic smooth neighbors. In Jian Guo and Ron Steinfeld, editors, *Advances in Cryptology – ASIACRYPT 2023*, pages 190–221, Singapore, 2023. Springer Nature Singapore. ISBN 978-981-99-8739-9. doi: 10.1007/978-981-99-8739-9_7.

[11] Jeffrey Burdges and Luca De Feo. Delay encryption. In Anne Canteaut and François-Xavier Standaert, editors, *Advances in Cryptology – EUROCRYPT 2021*, pages 302–326, Cham, 2021. Springer International Publishing. ISBN 978-3-030-77870-5. doi: 10.1007/978-3-030-77870-5_11.

[12] Shiping Cai, Mingjie Chen, and Christophe Petit. Faster algorithms for isogeny computations over extensions of finite fields. Cryptology ePrint Archive, Paper 2024/1852, 2024. URL `https://eprint.iacr.org/2024/1852`.

[13] Wouter Castryck, Tanja Lange, Chloe Martindale, Lorenz Panny, and Joost Renes. CSIDH: An efficient post-quantum commutative group action. In Thomas Peyrin and Steven Galbraith, editors, *Advances in Cryptology – ASIACRYPT 2018*, pages 395–427, Cham, 2018. Springer International Publishing. doi: 10.1007/978-3-030-03332-3_15.

[14] Jorge Chavez-Saab, Francisco Rodríguez-Henríquez, and Mehdi Tibouchi. Verifiable isogeny walks: Towards an isogeny-based postquantum VDF. In Riham AlTawy and Andreas Hülsing, editors, *Selected Areas in Cryptography*, pages 441–460, Cham, 2022. Springer International Publishing. doi: 10.1007/978-3-030-99277-4_21.

[15] Jorge Chavez-Saab, Maria Corte-Real Santos, Luca De Feo, Jonathan Komada Eriksen, Basil Hess, David Kohel, Antonin Leroux, Patrick Longa, Michael Meyer, Lorenz Panny, Sikhar Patranabis, Christophe Petit, Francisco Rodríguez Henríquez, Sina Schaeffler, and Benjamin Wesolowski. SQIsign algorithm specifications and supporting documentation. Project Homepage, 2023. URL `https://sqisign.org/spec/sqisign-20230601.pdf`.

[16] Peter Chvojka and Tibor Jager. Simple, fast, efficient, and tightly-secure non-malleable non-interactive timed commitments. In Alexandra Boldyreva and

Vladimir Kolesnikov, editors, *Public-Key Cryptography – PKC 2023*, pages 500–529, Cham, 2023. Springer Nature Switzerland. doi: 10.1007/978-3-031-31368-4_18.

[17] Maria Corte-Real Santos, Jonathan Komada Eriksen, Michael Meyer, and Krijn Reijnders. Aprèssqi: Extra fast verification for sqisign using extension-field signing. In Marc Joye and Gregor Leander, editors, *Advances in Cryptology – EUROCRYPT 2024*, pages 63–93, Cham, 2024. Springer Nature Switzerland. ISBN 978-3-031-58716-0. doi: 10.1007/978-3-031-58716-0_3.

[18] Craig Costello, Michael Meyer, and Michael Naehrig. Sieving for Twin Smooth Integers with Solutions to the Prouhet-Tarry-Escott Problem. In Anne Canteaut and François-Xavier Standaert, editors, *Advances in Cryptology – EUROCRYPT 2021*, pages 272–301, Cham, 2021. Springer International Publishing. ISBN 978-3-030-77870-5. doi: 10.1007/978-3-030-77870-5_10.

[19] Pierrick Dartois, Antonin Leroux, Damien Robert, and Benjamin Wesolowski. SQIsignHD: New dimensions in cryptography. In Marc Joye and Gregor Leander, editors, *Advances in Cryptology – EUROCRYPT 2024*, pages 3–32, Cham, 2024. Springer Nature Switzerland. ISBN 978-3-031-58716-0. doi: 10.1007/978-3-031-58716-0_1.

[20] Luca De Feo. Mathematics of isogeny based cryptography. Preprint, 2017. URL `https://arxiv.org/abs/1711.04062`.

[21] Luca De Feo, Simon Masson, Christophe Petit, and Antonio Sanso. Verifiable delay functions from supersingular isogenies and pairings. In Steven D. Galbraith and Shiho Moriai, editors, *Advances in Cryptology – ASIACRYPT 2019*, pages 248–277, Cham, 2019. Springer International Publishing. doi: 10.1007/978-3-030-34578-5_10.

[22] Luca De Feo, David Kohel, Antonin Leroux, Christophe Petit, and Benjamin Wesolowski. SQISign: Compact post-quantum signatures from quaternions and isogenies. In Shiho Moriai and Huaxiong Wang, editors, *Advances in Cryptology – ASIACRYPT 2020*, pages 64–93, Cham, 2020. Springer International Publishing. doi: 10.1007/978-3-030-64837-4_3.

[23] Luca De Feo, Antonin Leroux, Patrick Longa, and Benjamin Wesolowski. New algorithms for the deuring correspondence. In Carmit Hazay and Martijn Stam, editors, *Advances in Cryptology – EUROCRYPT 2023*, pages 659–690, Cham, 2023. Springer Nature Switzerland. doi: 10.1007/978-3-031-30589-4_23.

[24] Thomas Decru, Luciano Maino, and Antonio Sanso. Towards a quantum-resistant weak verifiable delay function. In Abdelrahaman Aly and Mehdi Tibouchi, editors, *Progress in Cryptology – LATINCRYPT 2023*, pages 149–168, Cham, 2023. Springer Nature Switzerland. doi: 10.1007/978-3-031-44469-2_8.

[25] Max Deuring. Die Typen der Multiplikatorenringe elliptischer funktionenkörper. *Abh. Math. Sem. Hansischen Univ.*, 14:197–272, 1941. doi: 10.1007/BF02940746.

[26] Max Duparc and Tako Boris Fouotsa. SQIPrime: A dimension 2 variant of SQISignHD with non-smooth challenge isogenies. In Kai-Min Chung and Yu Sasaki, editors, *Advances in Cryptology – ASIACRYPT 2024*, pages 396–429, Singapore, 2025. Springer Nature Singapore. ISBN 978-981-96-0891-1. doi: 10.1007/978-981-96-0891-1_13.

[27] Kirsten Eisenträger, Sean Hallgren, Kristin Lauter, Travis Morrison, and Christophe Petit. Supersingular isogeny graphs and endomorphism rings: Reductions and solutions. In Jesper Buus Nielsen and Vincent Rijmen, editors, *Advances in Cryptology – EUROCRYPT 2018*, pages 329–368, Cham, 2018. Springer International Publishing. doi: 10.1007/978-3-319-78372-7_11.

[28] Jonathan Komada Eriksen, Lorenz Panny, Jana Sotáková, and Mattia Veroni. Deuring for the people: Supersingular elliptic curves with prescribed endomorphism ring in general characteristic. Cryptology ePrint Archive, Paper 2023/106, 2023. URL `https://eprint.iacr.org/2023/106`.

[29] Lov K. Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing*, STOC '96, pages 212–219, New York, NY, USA, 1996. Association for Computing Machinery. doi: 10.1145/237814.237866.

[30] David Jao, Reza Azarderakhsh, Matthew Campagna, Craig Costello, Luca De Feo, Basil Hess, Amir Jalali, Brian Koziel, Brian LaMacchia, Patrick Longa, Michael Naehrig, Geovandro Pereira, Joost Renes, Vladimir Soukharev, and David Urbanik. Supersingular isogeny key encapsulation. Project Homepage, 2020. URL `https://sike.org/files/SIDH-spec.pdf`.

[31] Jonathan Katz, Julian Loss, and Jiayu Xu. On the security of time-lock puzzles and timed commitments. In Rafael Pass and Krzysztof Pietrzak, editors, *Theory of Cryptography*, pages 390–413, Cham, 2020. Springer International Publishing. doi: 10.1007/978-3-030-64381-2_14.

[32] David Kohel, Kristin Lauter, Christophe Petit, and Jean-Pierre Tignol. On the quaternion $\ell$-isogeny path problem. *LMS J. Comput. Math.*, 17:418–432, 2014. doi: 10.1112/S1461157014000151.

[33] Antonin Leroux. Verifiable random function from the deuring correspondence and higher dimensional isogenies. Springer-Verlag, 2025.

[34] Kohei Nakagawa, Hiroshi Onuki, Wouter Castryck, Mingjie Chen, Riccardo Invernizzi, Gioella Lorenzon, and Frederik Vercauteren. SQIsign2D-east: A new signature scheme using 2-dimensional isogenies. In Kai-Min Chung and Yu Sasaki, editors, *Advances in Cryptology – ASIACRYPT 2024*, pages 272–303, Singapore, 2025. Springer Nature Singapore. ISBN 978-981-96-0891-1. doi: 10.1007/978-981-96-0891-1_9.

[35] Hiroshi Onuki and Kohei Nakagawa. Ideal-to-isogeny algorithm using 2-dimensional isogenies and its application to sqisign. In Kai-Min Chung and Yu Sasaki, editors, *Advances in Cryptology – ASIACRYPT 2024*, pages 243–271, Singapore, 2025. Springer Nature Singapore. ISBN 978-981-96-0891-1. doi: 10.1007/978-981-96-0891-1_8.

[36] Christophe Petit and Spike Smith. An improvement to the quaternion analogue of the $\ell$-isogeny path problem. MathCrypt 2018, 2018. URL `https://crypto.iacr.org/2018/affevents/mathcrypt/medias/08-50_3.pdf`.

[37] Ronald L. Rivest, Adi Shamir, and David A. Wagner. Time-lock puzzles and timed-release crypto. Technical report, USA, 1996.

[38] Joseph H. Silverman. *The Arithmetic of Elliptic Curves*, volume 106 of *Graduate Texts in Mathematics*. Springer New York, 1986. doi: 10.1007/978-1-4757-1920-8.

[39] Sri Aravinda Krishnan Thyagarajan, Guilhem Castagnos, Fabian Laguillaumie, and Giulio Malavolta. Efficient CCA timed commitments in class groups. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, CCS '21, page 2663–2684, New York, NY, USA, 2021. Association for Computing Machinery. doi: 10.1145/3460120.3484773.

[40] John Voight. *Quaternion algebras*, volume 288 of *Graduate Texts in Mathematics*. Springer Cham, 2021. doi: 10.1007/978-3-030-56694-4.

[41] Jacques Vélu. Isogénies entre courbes elliptiques. *Comptes Rendus Hebdomadaires des Séances de l'Académie des Sciences, Série A*, 273, N°4:238–241, 1971.

[42] Benjamin Wesolowski. The supersingular isogeny path and endomorphism ring problems are equivalent. In *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1100–1111, 2022. doi: 10.1109/FOCS52979.2021.00109.