Timed Commitments and Timed Encryption: Generic Constructions and Instantiations from Isogenies

Mingjie Chen[1] and Jonas Meers[2]

[1] COSIC, KU Leuven, Belgium
mjchennn555@gmail.com
[2] Ruhr University Bochum, Germany
research@meers.org

Abstract. Introduced by Boneh and Naor (CRYPTO 2000), timed commitments are a versatile primitive that found numerous applications in e-voting, contract signing and auctions. In TCC 2020, Katz, Loss and Xu showed that non-interactive timed commitments (NITC) can be generically built from timed public key encryption (TPKE). Unfortunately, almost all constructions for either primitive rely on classical, i.e. non post-quantum, assumptions or require inefficient building blocks like indistinguishable obfuscation or fully homomorphic encryption.
In this work, we propose generic constructions for non-interactive timed commitments and timed encryption, assuming only efficient building blocks like verifiable random functions, trapdoor delay functions and NIZK proof systems. Both our NITC (called LEIBNITC) and our TPKE (called NYTPKE) can be instantiated from isogenies, making them post-quantum secure. The instantiation of LEIBNITC with isogenies is very efficient and yields commitments of size 2328 bits, representing one of the most efficient timed commitments in the literature.

1 Introduction

A Non-Interactive Timed Commitment (NITC) is a regular commitment with the added property that it can be opened by anyone after a certain delay t has passed. That is, the usual *hiding* property only holds up to time t and no guarantees are made afterwards. On the other hand, the commitment should be *binding* irrespective of the delay. Timed commitments have their roots in the seminal work by Rivest, Shamir and Wagner on the closely related concept of *time-lock puzzles* [34] and were subsequently introduced by Boneh and Naor [13].

Applications. Timed commitments were originally motivated by applications such as sealed-bid auctions, where their ability to enforce fairness is particularly valuable. Indeed, when implementing an auction with a regular commitment, dishonest parties might simply refuse to open their commitment after the bidding phase to avoid paying a high price. Timed commitments overcome this issue by allowing the auctioneer (or any other party) to *forcefully* open the commitment

on behalf of a dishonest party. This and other real-world applications like e-voting and contract signing contributed to the fact that timed commitments are a very active field of research [13, 17, 26, 28, 36].

Constructions of NITC. Constructing a NITC requires an inherently sequential operation to achieve the desired delay. To this end, many of the classical NITC constructions rely on repeated squaring, either over an RSA modulus N or in groups of unknown order like class groups [28, 36]. While the (classical) hardness of computing $g^{2^t} \bmod N$ via repeated squaring is well understood, any construction building on top of it is inherently not post-quantum secure. For example, the construction of Katz, Loss and Xu uses the factorization of N as a trapdoor, allowing them to quickly compute $g^{2^t} \bmod N$ in less than t steps. Indeed, it suffices to compute $x \equiv 2^t \bmod \phi(N)$ (which requires knowledge of the factorization of N) and subsequently $g^x \bmod N$. Furthermore, in the same work it is shown that the sequentiality of repeated squaring is *equivalent* to factoring the RSA modulus N in a strengthened version of the Algebraic Group Model [28]. Likewise, constructions based on repeated squaring in class groups suffer from a similar issue due to an algorithm by Biasse and Song that computes the class group structure in quantum polynomial time [9]. Alternative and plausibly post-quantum secure approaches exist, but these rely on indistinguishable obfuscation (iO) and are thus far from practical [10]. Therefore, as the transition to post-quantum secure cryptography continues, there does not exist a (practical) post-quantum secure NITC construction. In this work, we aim to overcome this limitation.

Isogenies. Within post-quantum cryptography, isogeny-based schemes have received considerable attention due to their very compact sizes. At its core, isogeny-based cryptography relies on the hardness of finding an isogeny $\varphi : E \to E'$ between two supersingular elliptic curves E, E' defined over $\mathbb{F}_{p^2}$. It is believed that the generic isogeny problem is hard to solve even for quantum computers. Isogenies saw their first major cryptographic use in the now broken SIDH key exchange by De Feo and Jao [27], with many protocols proposed in the following years [1, 4, 5, 6, 21]. A common limitation of isogeny-based constructions is their relatively low computational efficiency, particularly in comparison with lattice-based constructions. However, within the context of timed commitments this may be seen as an advantage since computations are intentionally designed to be time-consuming anyway.

1.1 Contributions

Our contributions can be summarized as follows:

- We give two *generic* NITC constructions, which can be instantiated from post-quantum secure primitives.
  - The first construction (called LEIBNITC) is a novel NITC construction based on a verifiable delay function (VDF) satisfying an information-theoretic property and a secret key encryption scheme. The construction is secure in the random oracle model.

- The second construction is a generalization of the construction of Katz, Loss and Xu [28], which is based on the Naor-Yung paradigm [32] and thus secure in the standard model. To this end, we first construct a *Timed PKE* (called NYTPKE), which can subsequently be transformed into a NITC. NYTPKE only requires a trapdoor delay function (TDF) that does not feature any verifiability.

– We give efficient instantiations of both constructions from isogenies, making them *post-quantum secure*. To this end, we show that the verifiable *random* function proposed by Leroux [30] gives rise to weak variants of both a verifiable *delay* function and a trapdoor delay function. "Weak" in this context means that the delay is *not* unconditional and depends on the amount of parallelism of the adversary (cf. Section 1.2).
– Instantiating LEIBNITC with isogenies yields very compact commitments. For a quantum security level of 128 bits, a delay of 2^{32}, parallelism 2^{16} and a message space of 256 bits, the commitment is as small as 2328 bits. In comparison, the (pre-quantum) NITC proposed in [17] features a commitment that is a factor 5.3× larger.

1.2 Technical Overview and Discussion

In the following discussion we mainly focus on our novel construction LEIBNITC. At its core, LEIBNITC uses a verifiable delay function VDF as well as a secret key encryption scheme SKE. To commit to a message, the committer first samples an instance x of VDF together with a solution y and a proof π. The commitment $\mathbf{C}$ is then computed as

$$k \leftarrow \mathsf{H}(x, y, \pi), \quad u \leftarrow \mathsf{SKE.Enc}(m; k), \quad \mathbf{C} \leftarrow (x, u),$$

where H is a random oracle.

Forcefully opening the commitment then requires solving the instance x, yielding a tuple (y', π'). Under the correctness of VDF we have $(y, \pi) = (y', \pi')$, allowing the decryption of the ciphertext u. The hiding property of LEIBNITC mainly reduces to the IND-CCA security of SKE, whereas the binding property relies on an information-theoretic uniqueness property of (y, π). We show that this information-theoretic property is indeed satisfied by our isogeny instantiation.

Further Properties of NITC. Apart from the aforementioned security guarantees, a timed commitment may satisfy two additional properties that are desirable in practice: committing to a value should be *fast* and opening the commitment (either honestly or forcefully) should be efficiently *verifiable*. To achieve the former, constructions typically involve a trapdoor during the commitment phase. To achieve the latter, a non-interactive proof can be used that asserts the correctness of the opening. We highlight that satisfying both properties simultaneously is non-trivial. For instance, the construction in [28] does not feature fast commitments since committing to a value takes as long as forcefully opening said commitment. Likewise, the construction in [7] has fast commitments but does

not feature any verifiability. In this work, we aim to achieve both properties with our LEIBNITC construction.[3]

Sequentiality of Isogeny Computations. When instantiated with isogenies, both our constructions rely on the sequentiality of computing the codomain of an isogeny $\varphi : E \to E'$ from a kernel generator $\langle K \rangle = \ker \varphi$. In the case where $T = \#\ker \varphi$ is a large prime, this computation typically involves enumerating the whole kernel, either via Vélu's formula [37] or via the $\sqrt{\text{élu}}$ algorithm [8], resulting in a computational complexity of $\mathcal{O}(T)$ and $\mathcal{O}(\sqrt{T})$, respectively. Unfortunately, Vélu's formula can be trivially parallelized, leading to a computational complexity of $\mathcal{O}(T/n + \log n)$ using n cores. Furthermore, it is shown in [16] that the $\sqrt{\text{élu}}$ algorithm can be parallelized as well, leading to a complexity of roughly $\mathcal{O}(\sqrt{T/n} + n)$. This amount of parallelization is undesirable for delay-based constructions and, in particular, timed commitments since parties with more resources can open a commitment early. However, in [11] Boneh et al. argue that any type of delay-based construction may suffer from (limited) parallelizability, including those based on repeated squaring and even repeated hashing. This lead the authors of [11] to define a relaxed notion of sequentiality where more parallelism is only beneficial up to a certain bound. Despite this, parameters in the pre-quantum setting can still be set more or less independently of the amount of parallelism a party possesses. Unfortunately, in the context of parallel Vélu an increase in parallelism will always improve its performance.[4]

To sidestep the issue, both LEIBNITC and NYTPKE require to choose a delay t *and* an upper bound on the parallelism n before parameter generation. More specifically, given an upper bound for n, the prime degree T of the isogeny is chosen in such a way that both the parallelized Vélu and $\sqrt{\text{élu}}$ strategy result in a computational complexity larger than t. Put differently, we do not aim for *unconditional* delays and instead let our parameters explicitly depend on n. Although this is suboptimal compared to pre-quantum constructions, such compromises may be expected when aiming for post-quantum security. Lastly, we highlight that both LEIBNITC and NYTPKE may be instantiated from different (post-quantum secure) primitives that satisfy a stronger sequentiality notion.

Comparison with [24]. Decru, Maino and Sanso propose another VDF in [24] based on the hardness of computing the codomains of isogenies of large prime degree T. In the following we denote their construction by DMS-VDF. Contrary to DeuringVDF proposed in Section 5.1, DMS-VDF relies on oriented elliptic curves over $\mathbb{F}_p$ where the prime p is chosen in such a way that on any curve $E/\mathbb{F}_p$ the kernels of the two unique horizontal T-isogenies are defined over $\mathbb{F}_{p^{T-1}}$. This way, even writing down the generators of their kernels takes exponential time, let alone computing the codomains, allowing the authors to argue for the sequentiality of DMS-VDF. Even though it is infeasible to write down the kernel of the two horizontal isogenies, it is possible to implicitly define them by their domain E, making it easy to create instances of DMS-VDF. Verifiability is

[3] Since NYTPKE is a generalization of [28] it does not feature fast commitments.
[4] The same is not true for $\sqrt{\text{élu}}$, see Section 5.3.

achieved by subsequently computing an HD-representation of the isogenies that can be evaluated efficiently. The downside of this approach, however, is that the prime p needs to be quite large: for a delay $t = 2^{32}$ we get $\log p = 2097152$, that is, p is a prime of size roughly 262 kilobytes [24, Section 3.2]. The required size of p stems from the fact that, in the $\mathbb{F}_p$-setting, a short equivalent isogeny can be found via lattice reductions. The prime therefore has to be chosen large enough such that lattice reductions cannot be performed in time less than t.

By contrast, elliptic curves in DeuringVDF are defined over $\mathbb{F}_{p^2}$, avoiding lattice reduction attacks and resulting in smaller parameters. This also makes it necessary to give the kernel generator explicitly as part of the instance. Verifiability is achieved in the same way as DMS-VDF. Lastly, we remark that DMS-VDF satisfies all properties required by LEIBNITC, including the information-theoretic uniqueness property (Definition 14). Furthermore, the fact that instances of DMS-VDF have to be sampled by a trusted party is not an issue, see Remark 2. Therefore, the more conservative DMS-VDF can also be used to instantiate LEIBNITC, although this results in larger commitments.

Detailed Comparison with Ahrens. In [2] Ahrens constructs a NITC (called SIGNITC) from isogenies using somewhat similar ideas to LEIBNITC. In particular, both constructions have the same sequentiality assumption. However, SIGNITC exhibits several important constraints compared to LEIBNITC. For instance, SIGNITC only fulfills an adapted and non-standard IND-CCA security notion [2, Definition 5.11]. The adapted IND-CCA game forbids any queries to the forced decommitment oracle[5] that would reveal the challenge message, severely limiting the attacker's capabilities. It is furthermore not established whether a reduction could even efficiently check that a query to FDecom would reveal the challenge message. In the same work it is also shown that the adapted IND-CCA notion is *necessary* by giving an efficient attacker against SIGNITC in the standard IND-CCA security game.

Furthermore, due to the lack of formal security reductions it is hard to estimate whether the security arguments for SIGNITC are *tight*. Indeed, NITCs as well as their underlying security assumptions impose explicit time bounds on the adversary, typically stating that it should be hard to solve a computational problem in a given and very concrete time t. However, it might happen that the security reduction itself is too inefficient and does not finish before time t, making it necessary to increase t in the assumption to account for that inefficiency.[6] Naturally, a tight reduction (both in terms of advantage and time) is preferable, yielding smaller parameters in the process. It is unclear whether there exists such a tight reduction for SIGNITC.

By contrast, our proposed schemes LEIBNITC and NYTPKE fulfill standard IND-CCA security notions without further constraints and feature a tight reduction with respect to both advantage and time.

[5] The FDecom oracle can be seen as an analogue to the Dec oracle in the PKE-version of IND-CCA.

[6] This is analogous to classical reductions where tightness usually refers to the advantage loss exhibited by the reduction.

1.3 Further Related Work

Several generic constructions for timed commitments exist. Freitag et al. construct a NITC from a VDF [26], but their constructions is only proven secure in the strong Auxiliary Input ROM and All-But-One string model. Furthermore, they do not distinguish between time-lock puzzles and NITCs. Ambrona et al. also give a generic NITC construction from a VDF in [3] that is somewhat similar to $\mathsf{LEIBNITC}$. However their construction only satisfies a passive security notion that is incomparable to the notions used in this work. Furthermore, achieving any form of active security is likely out of reach for their construction. Indeed, their construction computes a symmetrical encryption $\mathsf{SKE.Enc}(\tau||m;k)$ where τ is a trapdoor from which the symmetric key k can be recovered. Such a construction would require a circulant security argument akin to [12], which rarely achieves active security.

In terms of post-quantum secure NITC, the literature is sparse. Malavolta et al. as well as Brakerski et al. construct homomorphic time-lock puzzles from Fully Homomorphic Encryption (FHE) [15,31]. However, both practical requirements and security notions for homomorphic TLPs are orthogonal to those of NITCs, which feature non-malleable properties. Lastly, Chvojka et al. also rely on FHE to construct time-released encryption [18].

2 Preliminaries

We denote by $[\![\cdot]\!]$ a boolean test that returns 1 if the enclosed statement is true and 0 otherwise. A function $g(n)$ is $O(f(n))$ if there exist constants $c > 0$ and n_0 such that for all $n \geq n_0$, $g(n) \leq c \cdot f(n)$; A function $g(n)$ is $\Theta(f(n))$ if there exist constants $c_1, c_2 > 0$ and n_0 such that for all $n \geq n_0$, $c_1 \cdot f(n) \leq g(n) \leq c_2 \cdot f(n)$. Denote by $\mathsf{poly}(f(n))$ the set of functions that are polynomial in $f(n)$. The "soft-O" notation $\widetilde{O}$ is shorthand for $O(\mathsf{poly}(\log f(n))f(n))$, which hides poly-logarithmic factors in the asymptotic complexity. We write $f_{\text{in}} : \{0,1\}^{\lfloor \log(N) \rfloor} \to \mathbb{P}^1(\mathbb{Z}/N\mathbb{Z})$ an injective function for an integer N.

Further preliminaries concerning NIZK proof systems and isogenies can be found in Supplementary Material A.

2.1 Verifiable Delay Functions

We recall a slightly modified definition of a verifiable delay function from [11] (see also Remark 2).

Definition 1 (Verifiable Delay Function). *A* $(t_{\Delta}, t_{\mathsf{smpl}}, t_{\mathsf{vrfy}}, p_{\mathsf{proc}})$-verifiable delay function $\mathsf{VDF} = (\mathsf{VDF.Setup}, \mathsf{VDF.Sample}, \mathsf{VDF.Eval}, \mathsf{VDF.Verify})$ *is a set of algorithms as follows:*

- $\mathsf{pp} = (\mathsf{ek}, \mathsf{vk}) \xleftarrow{\$} \mathsf{VDF.Setup}$*: The randomized setup algorithm takes no input and produces a public parameter* pp *that consists of an evaluation key* ek *and a verification key* vk*. By convention, the public parameter specifies an input space* $\mathcal{X}$ *and an output space* $\mathcal{Y}$*.*

– $(x, y, \pi) \xleftarrow{\$} \mathsf{VDF.Sample}(\mathsf{pp})$*: The randomized sample algorithm takes as input a public parameter and outputs* $x \in \mathcal{X}$*,* $y \in \mathcal{Y}$ *and a (possibly empty) proof* π*.* VDF.Sample *is required to run in time* t_{smpl}*.*
– $(y, \pi) \leftarrow \mathsf{VDF.Eval}(\mathsf{ek}, x)$*: The deterministic evaluation algorithm takes as input an evaluation key* ek *and* $x \in \mathcal{X}$*. It produces an output* $y \in \mathcal{Y}$ *and a (possibly empty) proof* π*. For all* pp *generated by* VDF.Setup *and all* $x \in \mathcal{X}$*, algorithm* VDF.Eval(ek, x) *must run in parallel time* t_Δ *with* p_{proc} *processors.*
– $\{0, 1\} \leftarrow \mathsf{VDF.Verify}(\mathsf{vk}, x, y, \pi)$*: The deterministic verification algorithm takes as input a verification key* vk*,* $x \in \mathcal{X}$*,* $y \in \mathcal{Y}$ *and a proof* π *and outputs a bit. Algorithm* VDF.Verify *must run in total time* $t_{\mathsf{vrfy}} \ll t_\Delta$*.*

It is required that $\mathrm{Im}(\mathsf{VDF.Sample}) = \mathcal{X}$*, i.e. each element in* $\mathcal{X}$ *can be the output of* VDF.Sample*. Furthermore, we require that membership in* $\mathcal{X}$ *can be efficiently tested. Lastly* VDF *must satisfy Correctness (Definition 3).*

Remark 2. Contrary to [11] we do not require that the input space $\mathcal{X}$ is efficiently sampleable. Instead, we assume the existence of an algorithm VDF.Sample that samples an instance together with a correct evaluation result (y, π). This implies that instances $x \in \mathcal{X}$ have to be sampled by a trusted party, but for the application at hand (NITC) this is not an issue. Indeed, instances are sampled by the committer who already knows the message that is ought to be contained in the commitment. Furthermore, we do not make any requirements on the amount of allowed parallelism in the definition of a VDF.

Definition 3 (Correctness). *A verifiable delay function* VDF *is* correct *if for all parameters* $\mathsf{pp} = (\mathsf{ek}, \mathsf{vk}) \xleftarrow{\$} \mathsf{VDF.Setup}$*, and all instances* $(x, y, \pi) \xleftarrow{\$} \mathsf{VDF.Sample}(\mathsf{pp})$*, if* $(y', \pi') \leftarrow \mathsf{VDF.Eval}(\mathsf{ek}, x)$ *then we have* $(y, \pi) = (y', \pi')$ *and* $\mathsf{VDF.Verify}(\mathsf{vk}, x, y, \pi) = 1$*.*

We furthermore define *Soundness* and *Sequentiality.*

Definition 4 (Soundness). *A verifiable delay function* VDF *is* (ϵ, τ)-sound *if for all algorithms* $\mathcal{A}$ *that run in time* τ

$$\Pr\left[\begin{array}{c}\mathsf{VDF.Verify}(\mathsf{vk}, x, y', \pi') = 1 \\ (y, \pi) \leftarrow \mathsf{VDF.Eval}(\mathsf{ek}, x) \\ y' \neq y\end{array}\middle|\begin{array}{c}\mathsf{pp} = (\mathsf{ek}, \mathsf{vk}) \xleftarrow{\$} \mathsf{VDF.Setup} \\ (x, y', \pi') \leftarrow \mathcal{A}(\mathsf{pp})\end{array}\right] \leq \epsilon.$$

Definition 5 (Sequentiality). *Let* VDF *be a verifiable delay function and let* $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$ *be a pair of randomized algorithms such that* $\mathcal{A}_0$ *runs in total time at most* σ_{pre} *and* $\mathcal{A}_1$ *runs in parallel time at most* σ_{post} *on* π_{proc} *processors. We define the advantage of* $\mathcal{A}$ *winning* **SeqVDF** *as*

$$\mathsf{Adv}^{(\sigma_{\mathsf{pre}}, \sigma_{\mathsf{post}}, \pi_{\mathsf{proc}})\text{-}\mathbf{SeqVDF}}_{\mathsf{VDF}}(\mathcal{A}) \coloneqq \Pr\left[y' = y \middle| \begin{array}{c}\mathsf{pp} = (\mathsf{ek}, \mathsf{vk}) \xleftarrow{\$} \mathsf{VDF.Setup} \\ L \leftarrow \mathcal{A}_0(\mathsf{pp}) \\ (x, y, \pi) \xleftarrow{\$} \mathsf{VDF.Sample}(\mathsf{pp}) \\ y' \leftarrow \mathcal{A}_1(L, \mathsf{pp}, x)\end{array}\right].$$

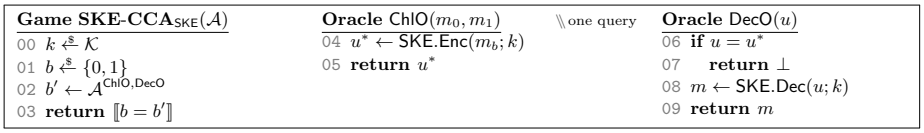

Fig. 1. Game defining **SKE-CCA** for a symmetric key encryption scheme SKE.

2.2 Symmetric Key Encryption

Definition 6 (Symmetric Key Encryption). *A* symmetric key encryption *scheme* $\mathsf{SKE} = (\mathsf{SKE.Enc}, \mathsf{SKE.Dec})$ *for some key space* $\mathcal{K}$ *is a tuple of algorithms as follows:*

- $u \leftarrow \mathsf{SKE.Enc}(m; k)$*: The deterministic encryption algorithm* SKE.Enc *takes as input a message* m *and a key* k *and outputs a ciphertext* u*.*
- $m \leftarrow \mathsf{SKE.Dec}(u; k)$*: The deterministic decryption algorithm* SKE.Dec *takes as input a ciphertext* u *and a key* k *and outputs a message* m*.*

SKE *is (perfectly) correct if for all* $k \in \mathcal{K}$ *and all messages* m *it holds* $m = \mathsf{SKE.Dec}(\mathsf{SKE.Enc}(m; k); k)$*.*

Definition 7 (SKE-CCA). *Let* SKE *be a symmetric key encryption scheme and consider the game* **SKE-CCA** *in Fig. 1. We define the advantage of an adversary* $\mathcal{A}$ *winning the* **SKE-CCA** *game as*

$$\mathsf{Adv}^{\mathbf{SKE\text{-}CCA}}_{\mathsf{SKE}}(\mathcal{A}) := \left| \Pr[\mathbf{SKE\text{-}CCA}_{\mathsf{SKE}}(\mathcal{A}) \Rightarrow 1] - \frac{1}{2} \right|.$$

2.3 Non-Interactive Timed Commitments

We recall the definition of a Non-Interactive Timed Commitment (NITC) from [28] as well as the corresponding security notions.

Definition 8 (Non-Interactive Timed Commitment). *A* $(t_{\mathsf{com}}, t_{\mathsf{cv}}, t_{\mathsf{dv}}, t_{\mathsf{fo}})$-Non-Interactive Timed Commitment *is a tuple of algorithms* $\mathsf{TC} = (\mathsf{TC.PGen}, \mathsf{TC.Com}, \mathsf{TC.ComVrfy}, \mathsf{TC.DecVrfy}, \mathsf{TC.FDecom})$ *with the following behavior:*

- $\mathsf{pp} \xleftarrow{\$} \mathsf{TC.PGen}$*: The randomized parameter generation algorithm* TC.PGen *takes no input and outputs a public parameter* pp*.*
- $(\mathbf{C}, \pi_{\mathsf{com}}, \pi_{\mathsf{dec}}) \xleftarrow{\$} \mathsf{TC.Com}(\mathsf{pp}, m)$*: The randomized commit algorithm* TC.Com *takes as input a public parameter* pp *and a message* $m \in \mathcal{M}$*. It outputs a commitment* $\mathbf{C}$ *and proofs* π_{com}*,* π_{dec} *in time at most* t_{com}*.*
- $\{0,1\} \leftarrow \mathsf{TC.ComVrfy}(\mathsf{pp}, \mathbf{C}, \pi_{\mathsf{com}})$*: The deterministic commitment verification algorithm* TC.ComVrfy *takes as input a public parameter* pp*, a commitment* $\mathbf{C}$ *and a proof* π_{com}*. It outputs 1 (accept) or 0 (reject) in time at most* t_{cv}*.*

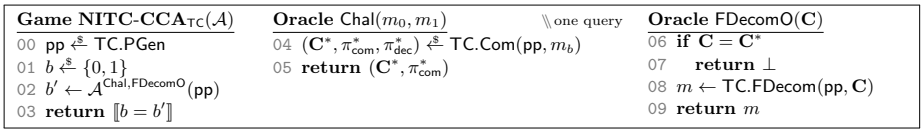

Fig. 2. Game defining **NITC-CCA** for a NITC scheme TC.

- $\{0,1\} \leftarrow \mathsf{TC.DecVrfy}(\mathsf{pp}, \mathbf{C}, m, \pi_{\mathsf{dec}})$ *The deterministic decommitment verification algorithm* TC.DecVrfy *takes as input a public parameter* pp*, commitment* $\mathbf{C}$*, message* m *and proof* π_{dec} *and outputs 1 (accept) or 0 (reject) in time at most* t_{dv}*.*
- $m \leftarrow \mathsf{TC.FDecom}(\mathsf{pp}, \mathbf{C})$*: The deterministic forced decommit algorithm* TC.FDecom *takes as input a public parameter* pp *and a commitment* $\mathbf{C}$ *and outputs a message* $m \in \mathcal{M}$ *or* $\perp$ *in time* at least t_{fo}*.*

We require that for all pp *output by* TC.PGen*, all* $m \in \mathcal{M}$ *and all* $(\mathbf{C}, \pi_{\mathsf{com}}, \pi_{\mathsf{dec}})$ *output by* $\mathsf{TC.Com}(\mathsf{pp}, m)$ *it holds that*

$$\mathsf{TC.ComVrfy}(\mathsf{pp}, \mathbf{C}, \pi_{\mathsf{com}}) = \mathsf{TC.DecVrfy}(\mathsf{pp}, \mathbf{C}, m, \pi_{\mathsf{dec}}) = 1$$

and $\mathsf{TC.FDecom}(\mathsf{pp}, \mathbf{C}) = m$*.*

We now define *indistinguishability* and *binding*. Compared to [28] our equivalent definition of indistinguishability allows a more fine-grained assessment of the runtime and parallelism of an adversary.

Definition 9 (NITC-CCA). *Let* TC *be a NITC scheme and let* $\mathcal{A}$ *be an algorithm with parallelism* ρ_{proc}*. Consider the game* **NITC-CCA** *defined in Fig. 2. We define the advantage of* $\mathcal{A}$ *winning* **NITC-CCA** *as*

$$\mathsf{Adv}^{(\tau_{\mathsf{pre}},\tau_{\mathsf{post}},\rho_{\mathsf{proc}})\text{-}\mathbf{NITC\text{-}CCA}}_{\mathsf{TC}}(\mathcal{A}) \coloneqq \left| \Pr[\mathbf{NITC\text{-}CCA}_{\mathsf{TC}}(\mathcal{A}) \Rightarrow 1] - \frac{1}{2} \right|.$$

Here τ_{pre} *and* τ_{post} *denote the maximum runtime of* $\mathcal{A}$ *before and after querying* Chal*, respectively.*

Definition 10 (NITC-BND). *Let* TC *be a NITC scheme and* $\mathcal{A}$ *be an algorithm. Consider the game* **NITC-BND** *defined in Fig. 3. We define the advantage of* $\mathcal{A}$ *winning the* **NITC-BND** *game as*

$$\mathsf{Adv}^{\mathbf{NITC\text{-}BND}}_{\mathsf{TC}}(\mathcal{A}) \coloneqq \Pr[\mathbf{NITC\text{-}BND}_{\mathsf{TC}}(\mathcal{A}) \Rightarrow 1].$$

Remark 11. Line 05 in Fig. 3 is not trivially prevented by the correctness of TC as the correctness is only guaranteed for commitments that were honestly sampled via TC.Com.

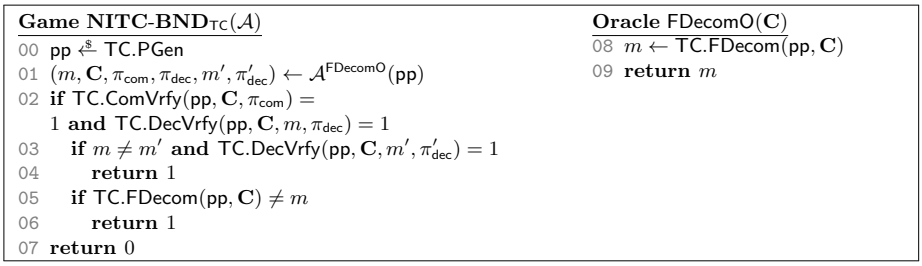

Fig. 3. Game defining **NITC-BND** for a NITC scheme TC.

2.4 Timed Public Key Encryption

Definition 12 (Timed Public-Key Encryption). *A* $(t_{\mathsf{e}}, t_{\mathsf{fd}}, t_{\mathsf{sd}})$-Timed Public-Key Encryption Scheme *is a tuple of algorithms* $\mathsf{TPKE} = (\mathsf{TPKE.KGen}, \mathsf{TPKE.Enc}, \mathsf{TPKE.Dec}_f, \mathsf{TPKE.Dec}_s)$ *with the following behavior:*

- $(\mathsf{pk}, \mathsf{sk}) \xleftarrow{\$} \mathsf{TPKE.KGen}$*: The randomized key-generation algorithm outputs a public and secret key pair.*
- $\mathsf{ct} \xleftarrow{\$} \mathsf{TPKE.Enc}(\mathsf{pk}, m)$*: The randomized encryption algorithm takes as input a public key* pk *and a message* m *and outputs a ciphertext* ct *in time at most* t_{e}*.*
- $m \leftarrow \mathsf{TPKE.Dec}_f(\mathsf{sk}, \mathsf{ct})$*: The deterministic fast decryption algorithm takes as input a secret key* sk *and a ciphertext* ct *and outputs a message* m *or* $\perp$ *in time at most* t_{fd}*.*
- $m \leftarrow \mathsf{TPKE.Dec}_s(\mathsf{pk}, \mathsf{ct})$*: The deterministic slow decryption algorithm takes as input a public key* pk *and a ciphertext* ct *and outputs a message* m *or* $\perp$ *in time* at least t_{sd}*.*

We require that for all $(\mathsf{pk}, \mathsf{sk}) \xleftarrow{\$} \mathsf{TPKE.KGen}$*, all* m *and all* $\mathsf{ct} \xleftarrow{\$} \mathsf{TPKE.Enc}(\mathsf{pk}, m)$ *it holds that* $\mathsf{TPKE.Dec}_f(\mathsf{sk}, \mathsf{ct}) = \mathsf{TPKE.Dec}_s(\mathsf{pk}, \mathsf{ct}) = m$*.*

Definition 13 (TPKE-CCA). *Let* TPKE *be a TPKE scheme and consider the game* **TPKE-CCA** *in Fig. 4. We define the advantage of an adversary* $\mathcal{A}$ *winning the* **TPKE-CCA** *as*

$$\mathsf{Adv}^{(\tau_{\mathsf{pre}}, \tau_{\mathsf{post}}, \rho_{\mathsf{proc}})\text{-}\mathbf{TPKE\text{-}CCA}}_{\mathsf{TPKE}}(\mathcal{A}) \coloneqq \left| \Pr[\mathbf{TPKE\text{-}CCA}_{\mathsf{TPKE}}(\mathcal{A}) \Rightarrow 1] - \frac{1}{2} \right|.$$

Here ρ_{proc} *denotes the amount of parallelism used by* $\mathcal{A}$*, while* τ_{pre} *and* τ_{post} *denote the runtime of* $\mathcal{A}$ *before and after querying* Chal*, respectively.*

2.5 Quaternion Algebras, Elliptic Curves, Isogenies and the Deuring Correspondence

Quaternion Algebras. Let p be a prime and let $\mathcal{B}_{p,\infty}$ denote the unique (up to isomorphism) quaternion algebra ramified precisely at p and ∞. We fix a $\mathbb{Q}$-basis $\langle 1, \mathbf{i}, \mathbf{j}, \mathbf{k} \rangle$ of $\mathcal{B}_{p,\infty}$ that satisfies $\mathbf{i}^2 = -q$, $\mathbf{j}^2 = -p$ and $\mathbf{k} = \mathbf{ij} = -\mathbf{ji}$ for some

Game TPKE-CCA$_{\mathsf{TPKE}}(\mathcal{A})$	**Oracle** $\mathsf{Chal}(m_0, m_1)$	**Oracle** $\mathsf{DecO}(\mathsf{ct})$
00 $(\mathsf{pk}, \mathsf{sk}) \xleftarrow{\$} \mathsf{TPKE.KGen}$	04 $\mathsf{ct}^* \xleftarrow{\$} \mathsf{TPKE.Enc}(\mathsf{pk}, m_b)$	06 **if** $\mathsf{ct} = \mathsf{ct}^*$
01 $b \xleftarrow{\$} \{0,1\}$	05 **return** ct^*	07 **return** $\perp$
02 $b' \leftarrow \mathcal{A}^{\mathsf{Chal},\mathsf{DecO}}(\mathsf{pk})$		08 $m \leftarrow \mathsf{TPKE.Dec}_f(\mathsf{sk}, \mathsf{ct})$
03 **return** $[\![b = b']\!]$		09 **return** m

Fig. 4. Game defining **TPKE-CCA** for a TPKE scheme TPKE.

integer q. A *fractional ideal* I in $B_{p,\infty}$ is a $\mathbb{Z}$-lattice of rank 4. We denote by $n(I)$ the *norm* of I as the largest rational number such that $n(\alpha) \in n(I)\mathbb{Z}$ for any $\alpha \in I$. An order $\mathcal{O}$ is a subring of $B_{p,\infty}$ that is also a fractional ideal. An order is called *maximal* when it is not contained in any other larger order. A fractional ideal is *integral* if it is contained in its *left order* $\mathcal{O}_L(I) = \{\alpha \in \mathcal{B}_{p,\infty} \mid \alpha I \subset I\}$, or equivalently in its *right order* $\mathcal{O}_R(I) = \{\alpha \in \mathcal{B}_{p,\infty} \mid I\alpha \subset I\}$. Two integral left $\mathcal{O}$-ideals I, J are said to be equivalent if and only if there exists $\alpha \in I$ such that $J = I\bar{\alpha}/n(I)$.

Supersingular Elliptic Curves and Isogenies. Let E, E_1, E_2 be elliptic curves defined over a finite field of characteristic p. The set of (isomorphism classes of) all such curves is denoted by $\mathcal{E}\ell\ell(\mathbb{F}_{p^2})$, each isomorphism class is uniquely represented by its j-invariant in $\mathbb{F}_{p^2}$. An isogeny from E_1 to E_2 is a non-constant rational map that is simultaneously a group homomorphism. An isogeny induces a field extension $K(E_1)/K(E_2)$ of function fields, and the isogeny is called separable, inseparable or purely inseparable if the extension of function field is of the respective type. The degree of the isogeny is the degree of the field extension $K(E_1)/K(E_2)$. The kernel of an isogeny $\phi : E_1 \to E_2$ is a finite subgroup of E_1. If the isogeny is separable, then the size of the kernel is equal to the degree of the isogeny. For every isogeny $\phi : E_1 \to E_2$ there exists a unique *dual* isogeny $\hat{\phi} : E_2 \to E_1$ such that $\deg(\phi) = \deg(\hat{\phi}) = d$ and $\phi \circ \hat{\phi} = [d]_{E_2}$ (and $\hat{\phi} \circ \phi = [d]_{E_1}$).

Let $\varphi : E \to E_1$, $\psi : E \to E_2$, and $\psi' : E_1 \to E_2$ be isogenies of coprime degrees. If $\ker \psi' = \varphi(\ker \psi)$ holds, we say that ψ' is the push-forward of ψ by φ and denote it by $\psi' = [\varphi]_*\psi$. Under the same situation, we say that ψ is the pull-back of ψ' by φ and denote it by $\psi = [\varphi]^*\psi$.

An isogeny from a curve E to itself is an *endomorphism*. The set $\mathrm{End}(E)$ of all endomorphisms of E forms a ring under addition and composition. $\mathrm{End}(E)$ is either an order in an imaginary quadratic field and E is called *ordinary*, or a maximal order in $\mathcal{B}_{p,\infty}$, in which case E is called *supersingular*.

The Deuring Correspondence. Fix a supersingular elliptic curve E, and an isomorphism $\iota : \mathcal{O} \simeq \mathrm{End}(E)$, there is a one to one correspondence between left $\mathcal{O}$-ideals I such that $(n(I), p) = 1$ and seperable outgoing isogenies from E up to post-composition by automorphisms of the codomain curve. Explicitly: given I an integral left $\mathcal{O}$-ideal such that $(n(I), p) = 1$, we define the *I-torsion subgroup* $E[I] = \{P \in E(\overline{\mathbb{F}}_{p^2}) : \iota(\alpha)(P) = 0 \text{ for all } \alpha \in I\}$. To I, we associate

the separable isogeny φ_I of kernel $E[I]$. Conversely given a separable isogeny φ, its *kernel ideal* is defined as $I_\varphi = \{\alpha \in \mathcal{O} \mid \iota(\alpha)(P) = 0 \text{ for all } P \in \ker(\varphi)\}$.

Algorithmic Building Blocks. To instantiate our constructions from isogenies in Section 5, we will need a few algorithms that are commonly used in isogeny-based cryptography. We now give brief descriptions of these algorithms.

- $\mathsf{MakeCanonical}(S)$ is an algorithm that selects the lexicographically smaller tuple in a set $S = \{(P, Q) \mid P, Q \in E[N]\}$ where E is an elliptic curve and N is an integer.
- $\mathsf{CanonicalBasis}(E, N)$ is a deterministic algorithm that computes a basis of $E[N]$.
- $\mathsf{IdealToIsogeny}(I, S, N)$ is an algorithm that takes a left $\mathcal{O}_0$ ideal I and a set S consisting of points on E_0, outputs the codomain E_I of the isogeny φ_I (determined by I under Deuring correspondence) and $\varphi_I(S)$. The last input is an optional input, and when given, it is an odd prime dividing $n(I)$ and the algorithm outputs a point that generates $\hat{\varphi}_I(E_I[N])$. The details of this algorithm are discussed in Supplementary Material B.
- $\mathsf{KaniDimFour}(E, (P, Q), F, (P', Q'), (c_1, c_2), S)$ is an algorithm that computes the unique isogeny φ from E to F of degree $2^a - c_1^2 - c_2^2$ that sends $(P, Q) \in E[2^{a+2}] \times E[2^{a+2}]$ to $(P', Q') \in F[2^{a+2}] \times F[2^{a+2}]$ and computes $\varphi(S)$ for a set S consisting of points on E. When it is clear what E is from the context, it is sometimes omitted, and (P, Q) is often taken to be the canonical basis of $E[2^{a+2}]$ in that case. See [20, Algorithm 6, 7] for more details of this algorithm.

3 LEIBNITC: A Generic NITC Construction

Let $\mathsf{VDF} = (\mathsf{VDF.Setup}, \mathsf{VDF.Sample}, \mathsf{VDF.Eval}, \mathsf{VDF.Verify})$ be a verifiable delay function. To construct our NITC protocol, we require the following additional properties.

Definition 14 (Unconditional Uniqueness). *A verifiable delay function* $\mathsf{VDF} = (\mathsf{VDF.Setup}, \mathsf{VDF.Sample}, \mathsf{VDF.Eval}, \mathsf{VDF.Verify})$ *satisfies* unconditional uniqueness *if for all (potentially unbounded) algorithms* $\mathcal{A}$ *we have*

$$\Pr\left[\begin{array}{c}(y', \pi') \neq (y, \pi) \\ \mathsf{VDF.Verify}(\mathsf{vk}, x, y', \pi') = 1\end{array}\middle|\begin{array}{c}\mathsf{pp} = (\mathsf{ek}, \mathsf{vk}) \xleftarrow{\$} \mathsf{VDF.Setup} \\ (x, y, \pi) \xleftarrow{\$} \mathsf{VDF.Sample}(\mathsf{pp}) \\ (y', \pi') \xleftarrow{\$} \mathcal{A}(\mathsf{pp}, x, y, \pi)\end{array}\right] = 0.$$

Compared to soundness (Definition 4), unconditional uniqueness also makes a statement about the uniqueness of the proof π. Furthermore, no requirement is imposed on the runtime of the algorithm, making the statement an information-theoretic property. Unconditional uniqueness relates to soundness in the following way.

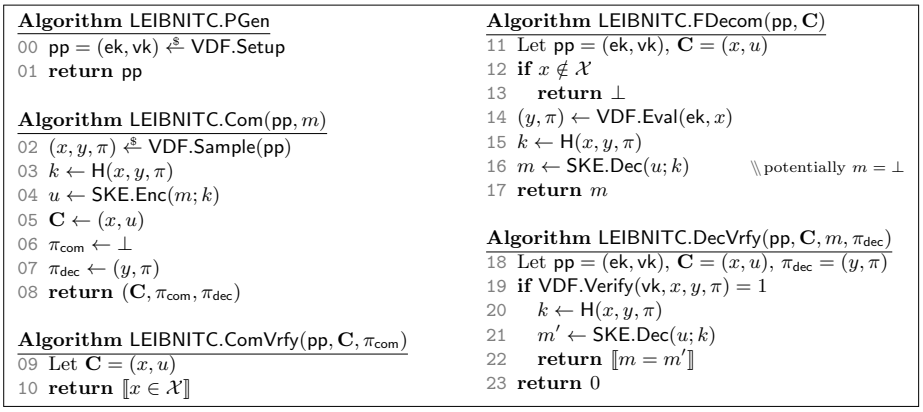

Fig. 5. Our Non-Interactive Timed Commitment scheme LEIBNITC[VDF, SKE, H].

Lemma 15. *If* VDF *satisfies unconditional uniqueness, then* VDF *is* $(0, \infty)$*-sound.*

Proof. Assume for the sake of contradiction that VDF is not $(0, \infty)$-sound. That is, there exists an (unbounded) adversary $\mathcal{A}$ that produces a verifying tuple (x, y', π') such that $y \neq y'$ where $(y, \pi) \leftarrow \mathsf{VDF.Eval}(\mathsf{ek}, x)$ with non-zero probability. Due to the fact that any $x \in \mathcal{X}$ can be output by VDF.Sample (as required by Definition 1), the above instance x also has a non-zero probability of being output by VDF.Sample. Furthermore, due to the correctness of VDF, algorithm VDF.Sample will output (x, y, π) identical to $\mathsf{VDF.Eval}(\mathsf{ek}, x)$. Hence, the honestly sampled instance $(x, y, \pi) \xleftarrow{\$} \mathsf{VDF.Sample}(\mathsf{pp})$ has a non-zero probability of being the same instance chosen by $\mathcal{A}$. Therefore, we can construct an adversary $\mathcal{B}$ that wins the unconditional uniqueness game in the following way: On input the honest instance (x, y, π), run $(\tilde{x}, y', \pi') \xleftarrow{\$} \mathcal{A}(\mathsf{pp})$ and abort if $\tilde{x} \neq x$. With non-zero probability, $x = \tilde{x}$ and $\mathcal{B}$ outputs (y', π'). Since $\mathcal{A}$ produces (y', π') such that $y \neq y'$ and $\mathsf{VDF.Verify}(\mathsf{vk}, x, y', \pi') = 1$, adversary $\mathcal{B}$ has a non-zero probability of winning the unconditional uniqueness game. □

We furthermore define *spreadness* as a measure for the min-entropy of instances generated by VDF.Sample.

Definition 16 (γ-Spreadness). *We say that a verifiable delay function* $\mathsf{VDF} = (\mathsf{VDF.Setup}, \mathsf{VDF.Sample}, \mathsf{VDF.Eval}, \mathsf{VDF.Verify})$ *has* γ-spreadness *if for all* $\mathsf{pp} \xleftarrow{\$} \mathsf{VDF.Setup}$ *and all* $x \in \mathcal{X}$ *we have*

$$\Pr\left[x = x^* \,\middle|\, (x^*, \cdot, \cdot) \xleftarrow{\$} \mathsf{VDF.Sample}(\mathsf{pp})\right] \leq \gamma.$$

We now introduce our new NITC scheme LEIBNITC[7] depicted in Fig. 5. It requires a verifiable delay function VDF with unconditional uniqueness, γ-

[7] Standing for "LargE Isogeny Based Non Interactive Timed Commitment" based on the isogeny instantiation in Section 5.

spreadness as well as a symmetric encryption scheme $\mathsf{SKE} = (\mathsf{SKE.Enc}, \mathsf{SKE.Dec})$ satisfying IND-CCA security and a random oracle H.

Lemma 17. *The* LEIBNITC *scheme depicted in Fig. 5 is correct. That is, for all parameters* $\mathsf{pp} \xleftarrow{\$} \mathsf{LEIBNITC.PGen}$*, all* $m \in \mathcal{M}$ *and all* $(\mathbf{C}, \pi_{\mathsf{com}}, \pi_{\mathsf{dec}}) \xleftarrow{\$} \mathsf{LEIBNITC.Com}(\mathsf{pp}, m)$ *it holds that*

$$\mathsf{LEIBNITC.ComVrfy}(\mathsf{pp}, \mathbf{C}, \pi_{\mathsf{com}}) = \mathsf{LEIBNITC.DecVrfy}(\mathsf{pp}, \mathbf{C}, m, \pi_{\mathsf{dec}}) = 1$$

and $\mathsf{LEIBNITC.FDecom}(\mathsf{pp}, \mathbf{C}) = m$.

Proof. This easily follows from the definition of LEIBNITC and the corresponding correctness properties of VDF and SKE. □

3.1 Security

We now prove the security of LEIBNITC. In particular, we prove indistinguishability of commitments and that commitments are *perfectly* binding. We start by proving the IND-CCA security of LEIBNITC.

Theorem 18. *Let* VDF *be a* $(t_\Delta, t_{\mathsf{smpl}}, t_{\mathsf{vrfy}}, p_{\mathsf{proc}})$*-verifiable delay function with unconditional uniqueness and* γ*-spreadness, and let* SKE *be a symmetric encryption scheme. For every adversary* $\mathcal{A}$ *against the* $(\tau_{\mathsf{pre}}, \tau_{\mathsf{post}}, \rho_{\mathsf{proc}})$-**NITC-CCA** *security of* $\mathsf{LEIBNITC}[\mathsf{VDF}, \mathsf{SKE}, \mathsf{H}]$ *there exist adversaries* $\mathcal{B}$ *and* $\mathcal{C}$ *such that*

$$\begin{aligned}\mathsf{Adv}^{(\tau_{\mathsf{pre}},\tau_{\mathsf{post}},\rho_{\mathsf{proc}})\text{-}\mathbf{NITC\text{-}CCA}}_{\mathsf{LEIBNITC}[\mathsf{VDF},\mathsf{SKE},\mathsf{H}]}(\mathcal{A}) \leq\ & \mathsf{Adv}^{(\sigma_{\mathsf{pre}},\sigma_{\mathsf{post}},\pi_{\mathsf{proc}})\text{-}\mathbf{SeqVDF}}_{\mathsf{VDF}}(\mathcal{B}) \\ & + 2 \cdot \mathsf{Adv}^{\mathbf{SKE\text{-}CCA}}_{\mathsf{SKE}}(\mathcal{C}) + (q_{\mathsf{FDecom}} + q_{\mathsf{H}}) \cdot \gamma.\end{aligned}$$

Here q_{FDecom} *and* q_{H} *denote the number of queries that* $\mathcal{A}$ *makes to* FDecomO *and* H*, respectively. Furthermore, we have*

$$\sigma_{\mathsf{pre}} = (q_{\mathsf{FDecom}} + q_{\mathsf{H}})^2 \cdot t_{\mathsf{vrfy}} + \tau_{\mathsf{pre}}, \qquad \pi_{\mathsf{proc}} = \rho_{\mathsf{proc}},$$
$$\sigma_{\mathsf{post}} = (q_{\mathsf{FDecom}} + q_{\mathsf{H}})^2 \cdot t_{\mathsf{vrfy}} + \tau_{\mathsf{post}}.$$

Proof. We prove the theorem by a sequence of games depicted in Fig. 6.

Game G_0. We start with the original $(\tau_{\mathsf{pre}}, \tau_{\mathsf{post}}, \rho_{\mathsf{proc}})$-**NITC-CCA** game for a non-interactive timed commitment.

$$\mathsf{Adv}^{(\tau_{\mathsf{pre}},\tau_{\mathsf{post}},\rho_{\mathsf{proc}})\text{-}\mathbf{NITC\text{-}CCA}}_{\mathsf{LEIBNITC}[\mathsf{VDF},\mathsf{SKE},\mathsf{H}]}(\mathcal{A}) = \left|\Pr\left[\mathsf{G}_0^{\mathcal{A}} \Rightarrow 1\right] - \frac{1}{2}\right|.$$

Game G_1. This game is identical to the original game, except that the game aborts during the computation of the challenge when $(x^*, \cdot, \cdot, \cdot) \in \mathcal{L}_{\mathsf{H}}$, i.e. x^* has already been queried to the random oracle H. Note that an entry gets added to $\mathcal{L}_{\mathsf{H}}$ either during a call to FDecomO or during a direct call to H. Due to the spreadness of VDF we therefore get

$$\left|\Pr\left[\mathsf{G}_0^{\mathcal{A}} \Rightarrow 1\right] - \Pr\left[\mathsf{G}_1^{\mathcal{A}} \Rightarrow 1\right]\right| \leq (q_{\mathsf{FDecom}} + q_{\mathsf{H}}) \cdot \gamma.$$

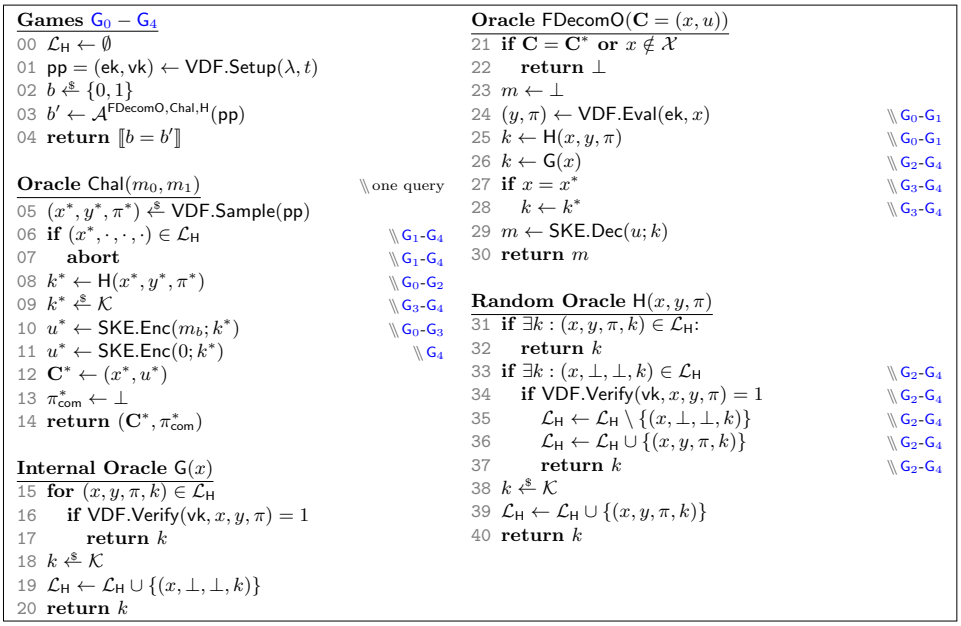

Fig. 6. Games $\mathsf{G}_0 - \mathsf{G}_4$ for the proof of Theorem 18.

Game G_2. In this game, the computation of (y, π) in $\mathsf{FDecomO}$ is skipped. Furthermore, the computation of the key k is handled by an auxiliary internal oracle G that only takes the VDF instance x as input. For each query x where there does not already exist a verifying tuple (x, y, π, k) in $\mathcal{L}_\mathsf{H}$, oracle G appends an entry $(x, \perp, \perp, k)$ to $\mathcal{L}_\mathsf{H}$ without specifying the corresponding $(y, \pi) = \mathsf{VDF.Eval}(\mathsf{ek}, x)$. To keep $\mathcal{L}_\mathsf{H}$ consistent, the random oracle H is patched whenever H is queried on the verifying tuple (x, y, π) (which furthermore can be efficiently checked via $\mathsf{VDF.Verify}$). Evidently, this change is only syntactical, thus

$$\left|\Pr\left[\mathsf{G}_1^{\mathcal{A}} \Rightarrow 1\right] - \Pr\left[\mathsf{G}_2^{\mathcal{A}} \Rightarrow 1\right]\right| = 0.$$

Game G_3. This is identical to the last game, except that a random challenge key k^* is used that does not come from the query (x^*, y^*, π^*) to the random oracle H.

Claim. This change is only detected if adversary $\mathcal{A}$ is able to compute y^* in time at most τ_{post}. Put differently, there exists an adversary $\mathcal{B}$ against **SeqVDF** such that

$$\left|\Pr\left[\mathsf{G}_2^{\mathcal{A}} \Rightarrow 1\right] - \Pr\left[\mathsf{G}_3^{\mathcal{A}} \Rightarrow 1\right]\right| \leq \mathsf{Adv}_{\mathsf{VDF}}^{(\sigma_{\mathsf{pre}}, \sigma_{\mathsf{post}}, \pi_{\mathsf{proc}})\text{-}\mathbf{SeqVDF}}(\mathcal{B}),$$

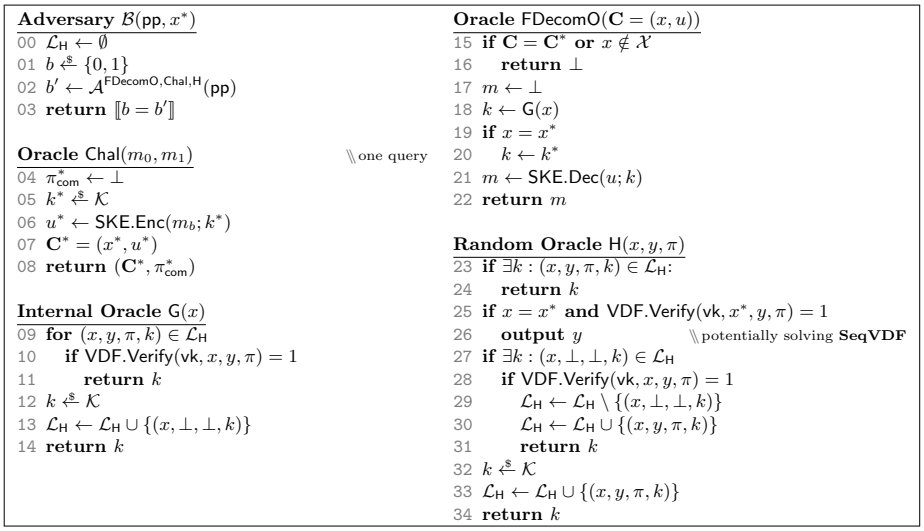

Fig. 7. Adversary $\mathcal{B}$ against **SeqVDF** used during the proof of Theorem 18.

where

$$\sigma_{\mathsf{pre}} \leq (q_{\mathsf{FDecom}} + q_{\mathsf{H}})^2 \cdot t_{\mathsf{vrfy}} + \tau_{\mathsf{pre}}, \qquad \pi_{\mathsf{proc}} = \rho_{\mathsf{proc}}$$
$$\sigma_{\mathsf{post}} \leq (q_{\mathsf{FDecom}} + q_{\mathsf{H}})^2 \cdot t_{\mathsf{vrfy}} + \tau_{\mathsf{post}}.$$

Proof (Claim). We formally construct adversary $\mathcal{B}$ in Fig. 7. Adversary $\mathcal{B}$ simulates the $\mathsf{FDecomO}$ oracle as in G_2. In particular, this simulation is efficient as $\mathcal{B}$ does not need to run $\mathsf{VDF.Eval}$ anymore. Instead, the evaluation is replaced by a lookup in the random oracle. For each lookup in the random oracle, $\mathcal{B}$ needs to run $\mathsf{VDF.Verify}$ at most $|\mathcal{L}_{\mathsf{H}}|$-many times. An entry can only be added to $\mathcal{L}_{\mathsf{H}}$ during a call to $\mathsf{FDecomO}$ or H, hence $|\mathcal{L}_{\mathsf{H}}| \leq q_{\mathsf{FDecom}} + q_{\mathsf{H}}$. Therefore, each of the at most $|\mathcal{L}_{\mathsf{H}}|$ lookups has a runtime of at most $|\mathcal{L}_{\mathsf{H}}| \cdot t_{\mathsf{vrfy}}$. It then embeds its own **SeqVDF** challenge x^* into the challenge commitment $\mathbf{C}^*$. Recall that since game G_1, it is guaranteed that x^* was not queried to the random oracle before, making the embedding possible. Next, $\mathcal{B}$ chooses a random k^* for the symmetric encryption. Crucially, k^* is correctly distributed until $\mathcal{A}$ queries the random oracle H on the tuple (x^*, y^*, π^*) where $(y^*, \pi^*) = \mathsf{VDF.Eval}(\mathsf{ek}, x^*)$. This query can however be detected via the $\mathsf{VDF.Verify}$ algorithm. Thus, as soon as $\mathcal{A}$ queries H on (x^*, y^*, π^*), adversary $\mathcal{B}$ is able to output y^* to the **SeqVDF** game. It is easy to see that if $\mathcal{A}$ is able to query H on that tuple in time τ_{post} using parallelism ρ_{proc}, then $\mathcal{B}$ was able to solve **SeqVDF** in time at most $(q_{\mathsf{FDecom}}+q_{\mathsf{H}})^2 \cdot t_{\mathsf{vrfy}} + \tau_{\mathsf{post}}$ using parallelism ρ_{proc}. (The argument for the preprocessing time is analogous.) In the other case where $\mathcal{A}$ does not query H on (x^*, y^*, π^*) the two games are indistinguishable as k^* is distributed identically in both games. □

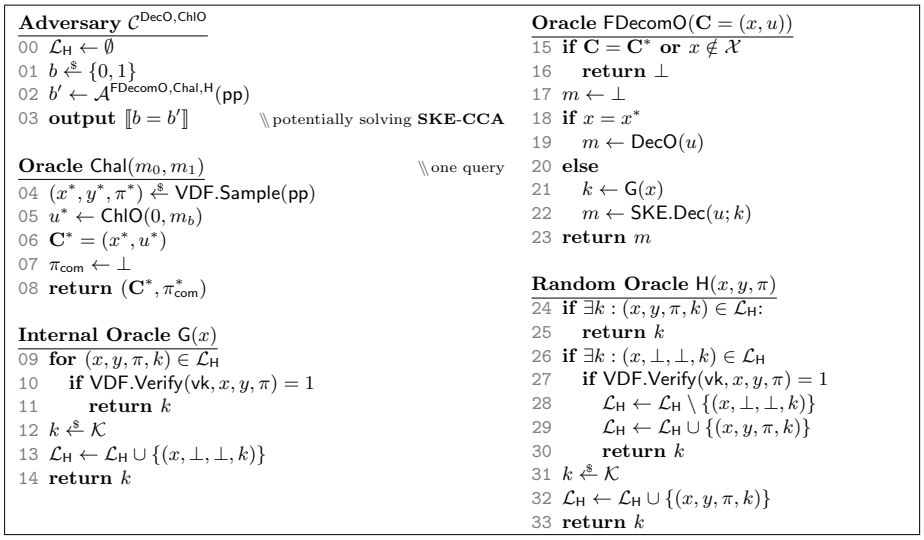

Fig. 8. Adversary $\mathcal{C}$ against **SKE-CCA** used during the proof of Theorem 18.

Game G_4. This is identical to the last game, except that during the challenge generation the message 0 is encrypted instead of m_b.

Claim. This change is only detected if $\mathcal{A}$ is able to win the **SKE-CCA** game of SKE. Thus, there exists an adversary $\mathcal{C}$ such that

$$\left|\Pr\left[\mathsf{G}_3^{\mathcal{A}} \Rightarrow 1\right] - \Pr\left[\mathsf{G}_4^{\mathcal{A}} \Rightarrow 1\right]\right| \leq 2 \cdot \mathsf{Adv}^{\mathbf{SKE\text{-}CCA}}_{\mathsf{SKE}}(\mathcal{C}).$$

Proof (Claim). Adversary $\mathcal{C}$ is formally constructed in Fig. 8. During the computation of the challenge commitment, adversary $\mathcal{C}$ uses its own ChlO oracle of the **SKE-CCA** game to get an encryption u^* of either 0 or m_b. Adversary $\mathcal{C}$ then includes u^* in the commitment $\mathbf{C}^*$, which is then sent to adversary $\mathcal{A}$. Note that due to the previous game hop the random oracle H is not queried on input (x^*, y^*, π^*), making the embedding of u^* in $\mathbf{C}^*$ possible. Furthermore, we highlight that $\mathcal{C}$ needs to output 1 iff u^* is an encryption of m_b in order to win the **SKE-CCA** game.

Subsequently, $\mathcal{C}$ simulates all other oracles as in the previous game. In case the $\mathsf{FDecomO}$ oracle is queried on a commitment (x^*, v) adversary $\mathcal{C}$ uses its own decryption oracle DecO to decrypt the ciphertext v. After receiving the bit b', adversary $\mathcal{C}$ outputs 1 iff $b = b'$. Indeed, if u' is an encryption of m_b, then $\mathcal{C}$ simulates G_3 perfectly and will win the **SKE-CCA** game with probability $\Pr[\mathsf{G}_3 \Rightarrow 1]$. On the other hand, if u^* is an encryption of 0, then $\mathcal{C}$ wins the **SKE-CCA** game with probability $1 - \Pr[\mathsf{G}_4 \Rightarrow 1]$ (note that $\mathcal{C}$ needs to output

0 to indicate that the message 0 was encrypted). Therefore

$$\begin{aligned}\mathsf{Adv}^{\mathbf{SKE\text{-}CCA}}_{\mathsf{SKE}}(\mathcal{C}) &= \left|\Pr\left[\mathcal{C} \text{ wins}\right] - \frac{1}{2}\right| \\ &\geq \left|\frac{1}{2}\cdot\Pr[\mathsf{G}_3 \Rightarrow 1] + \frac{1}{2}\cdot(1-\Pr[\mathsf{G}_4 \Rightarrow 1]) - \frac{1}{2}\right| \\ &= \left|\frac{1}{2}\cdot(\Pr[\mathsf{G}_3 \Rightarrow 1] - \Pr[\mathsf{G}_4 \Rightarrow 1])\right|.\end{aligned}$$

□

Finally, in G_4 the challenge commitment $\mathbf{C}^*$ is completely independent of the choice of b. Therefore we have

$$\Pr[\mathsf{G}_4^{\mathcal{A}} \Rightarrow 1] = \frac{1}{2}.$$

Collecting all the probabilities yields the desired result. □

We now turn to the NITC-BND security of LEIBNITC. Since we require that the underlying VDF satisfies unconditional uniqueness, we get *perfect* binding. That is, not even an information-theoretic adversary can win the NITC-BND game with non-zero probability.

Theorem 19. *Let* VDF *be a verifiable delay function with unconditional uniqueness and let* SKE *be a symmetric encryption scheme. There exist no adversary* $\mathcal{A}$ *against* LEIBNITC[VDF, SKE, H] *such that*

$$\mathsf{Adv}^{\mathbf{NITC\text{-}BND}}_{\mathsf{LEIBNITC}[\mathsf{VDF},\mathsf{SKE},\mathsf{H}]}(\mathcal{A}) > 0.$$

Proof. Assume that the output of adversary $\mathcal{A}$ is $(m, \mathbf{C}, \pi_{\mathsf{com}}, \pi_{\mathsf{dec}}, m', \pi'_{\mathsf{dec}})$ and that

$$\mathsf{LEIBNITC.ComVrfy}(\mathsf{pp}, \mathbf{C}, \pi_{\mathsf{com}}) = \mathsf{LEIBNITC.DecVrfy}(\mathsf{pp}, \mathbf{C}, m, \pi_{\mathsf{dec}}) = 1$$

(otherwise there is nothing to prove). Furthermore, let $\mathbf{C} = (x, u)$, $\pi_{\mathsf{dec}} = (y, \pi)$ and $\pi'_{\mathsf{dec}} = (y', \pi')$ as per definition of LEIBNITC.

We start by assuming that the adversary satisfies the first winning condition in Line 03 of Fig. 3, that is,

$$m \neq m' \quad \wedge \quad \mathsf{LEIBNITC.DecVrfy}(\mathsf{pp}, \mathbf{C}, m', \pi'_{\mathsf{dec}}) = 1.$$

There are now two cases.

Case 1: $\pi_{\mathsf{dec}} = \pi'_{\mathsf{dec}}$. In this case LEIBNITC.DecVrfy will internally compute the same key k for the symmetric decryption as $k = \mathsf{H}(x, y, \pi) = \mathsf{H}(x, y', \pi')$. Since the symmetric decryption is a deterministic algorithm, it will yield the same message $m = \mathsf{SKE.Dec}(u; k)$ and thus the final check in Line 22 of Fig. 5 will fail for message $m' \neq m$. We conclude that this case cannot occur.

Case 2: $\pi_{\mathsf{dec}} \neq \pi'_{\mathsf{dec}}$. Recall that LEIBNITC.DecVrfy internally calls VDF.Verify to assert the validity of π'_{dec}. Due to the unconditional uniqueness property of VDF we have

$$[\mathsf{VDF.Verify}(\mathsf{vk}, x, y, \pi) = \mathsf{VDF.Verify}(\mathsf{vk}, x, y', \pi') = 1] \quad \Rightarrow \quad (y, \pi) = (y', \pi').$$

In particular, this also holds for adversarially chosen instances $x \in \mathcal{X}$. Indeed, due to the definition of VDF each instance $x \in \mathcal{X}$ chosen by $\mathcal{A}$ can be the output of VDF.Sample and membership in $\mathcal{X}$ can efficiently be tested (which is checked during LEIBNITC.ComVrfy). Hence, if any weak instances $x \in \mathcal{X}$ would exist that imply $(y, \pi) \neq (y', \pi')$, then they have a non-zero chance of being chosen by an honest evaluation of VDF.Sample. The definition of unconditional uniqueness, however, asserts a success probability of 0, eliminating the existence of such weak instances that could be maliciously generated by $\mathcal{A}$. We therefore conclude that this case cannot occur as well since necessarily $\mathsf{VDF.Verify}(\mathsf{vk}, x, y', \pi') \neq 1$.

We now assume that adversary $\mathcal{A}$ satisfied the second winning condition in Line 05 of Fig. 3, that is,

$$\mathsf{LEIBNITC.FDecom}(\mathsf{pp}, \mathbf{C}) \neq m.$$

With the same argument as before, we know that any VDF instance x output by $\mathcal{A}$ can also be the output of an honest execution of VDF.Sample. However, the correctness of LEIBNITC (Lemma 17) then implies that necessarily $\mathsf{LEIBNITC.FDecom}(\mathsf{pp}, \mathbf{C}) = m$, leading to a contradiction. □

4 NYTPKE: A Construction Based on KLX

In this section we construct another NITC scheme by generalizing the approach by Katz, Loss and Xu (KLX) [28]. To this end, we first construct a *Timed Public Key Encryption* (TPKE) scheme from a *Trapdoor Delay Function* and then use the transformation in [28] to construct a NITC. Notably, by using the Naor-Yung paradigm [32] this construction does not rely on random oracles and is thus secure in the *standard* model.

We begin by introducing the notion of a trapdoor delay function (TDF).

Definition 20 (Trapdoor Delay Function). *A* trapdoor delay function $(t_\Delta, t_{\mathsf{td}}, p_{\mathsf{proc}})$-$\mathsf{TDF} = (\mathsf{TDF.Setup}, \mathsf{TDF.Eval})$ *is a set of algorithms as follows:*

- $(\mathsf{ek}, \mathsf{td}) \xleftarrow{\$} \mathsf{TDF.Setup}$*: The randomized setup algorithm produces a public evaluation key* ek *as well as a trapdoor* td*. By convention the evaluation key specifies an input space* $\mathcal{X}$ *and an output space* $\mathcal{Y}$*.*
- $y \leftarrow \mathsf{TDF.Eval}(\mathsf{ek}, x, \mathsf{td})$*: The deterministic evaluation algorithm takes an evaluation key* ek*, some* $x \in \mathcal{X}$ *and a (possibly empty) trapdoor as input. It produces an output* $y \in \mathcal{Y}$*. For all* $(\mathsf{ek}, \mathsf{td}) \xleftarrow{\$} \mathsf{TDF.Setup}$ *and all* $x \in \mathcal{X}$ *it is required that* $\mathsf{TDF.Eval}(\mathsf{ek}, x, \mathsf{td})$ *terminates in total time* t_{td}*. Similarly,* $\mathsf{TDF.Eval}(\mathsf{ek}, x, \perp)$ *must run in parallel time* t_Δ *with parallelism* p_{proc}*.*

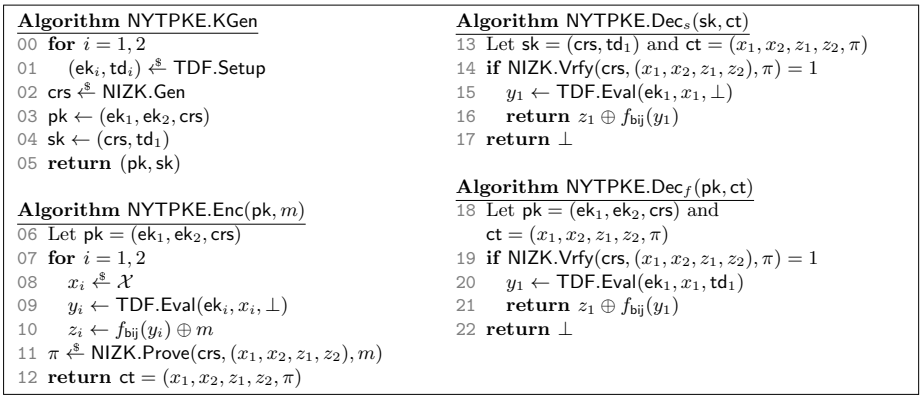

Fig. 9. Our Timed Public-Key Encryption scheme $\mathsf{NYTPKE}[\mathsf{TDF}, \mathsf{NIZK}]$.

It is required that $\mathcal{X}$ and $\mathcal{Y}$ are efficiently sampleable. Furthermore, TDF *has to satisfy correctness (Definition 21). We do not require any verifiability of the output.*

Definition 21 (Correctness). *A trapdoor delay function* TDF *is* correct *if for all* $(\mathsf{ek}, \mathsf{td}) \xleftarrow{\$} \mathsf{TDF.Setup}$ *and all* $x \in \mathcal{X}$ *we have* $\mathsf{TDF.Eval}(\mathsf{ek}, x, \bot) = \mathsf{TDF.Eval}(\mathsf{ek}, x, \mathsf{td})$.

We now present our construction NYTPKE[8], which can be found in Fig. 9. It requires a TDF, a bijective map $f_{\mathsf{bij}} : \mathcal{Y} \to \{0,1\}^{|m|}$ as well as a simulation-sound NIZK for the relation

$$\mathcal{R} = \left\{ \left((\mathsf{ek}_1, \mathsf{ek}_2, x_1, x_2, z_1, z_2), m \right) \middle| \bigwedge_{i=1,2} z_i = f_{\mathsf{bij}}(\mathsf{TDF.Eval}(\mathsf{ek}_i, x_i, \bot)) \oplus m \right\}$$

featuring fast verification and simulation. Note that we do not require that f_{bij}^{-1} is efficiently computable.

4.1 Security

Similar to [28] we will need the following decisional assumption.

Definition 22 (Sequential Indistinguishability). *Let $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$ be a pair of randomized algorithms such that $\mathcal{A}_0$ runs in total time σ_{pre} and $\mathcal{A}_1$ runs in parallel time σ_{post} on at most π_{proc} processors. For a trapdoor delay function* TDF *we define the advantage of $\mathcal{A}$ winning the $(\sigma_{\mathsf{pre}}, \sigma_{\mathsf{post}}, \pi_{\mathsf{proc}})$-***SeqDist** *game*

[8] Pronounced "nitpick".

as

$$\mathsf{Adv}^{(\sigma_{\mathsf{pre}},\sigma_{\mathsf{post}},\pi_{\mathsf{proc}})\text{-}\mathbf{SeqDist}}_{\mathsf{TDF}}(\mathcal{A}) \coloneqq \left|\Pr\left[b' = b \;\middle|\; \begin{array}{c} b \xleftarrow{\$} \{0,1\} \\ (\mathsf{ek},\mathsf{td}) \xleftarrow{\$} \mathsf{TDF.Setup} \\ L \leftarrow \mathcal{A}_0(\mathsf{ek}) \\ x \xleftarrow{\$} \mathcal{X} \\ y_0 \leftarrow \mathsf{TDF.Eval}(\mathsf{ek}, x, \mathsf{td}) \\ y_1 \xleftarrow{\$} \mathcal{Y} \\ b' \leftarrow \mathcal{A}_1(L, \mathsf{ek}, x, y_b) \end{array}\right] - \frac{1}{2}\right|.$$

The security of NYTPKE is summarized in the following theorem.

Theorem 23. *Let* NIZK *be a* $(t_{\mathsf{prv}}, t_{\mathsf{vrfy}}, t_{\mathsf{sgen}}, t_{\mathsf{sprv}})$*-NIZK proof system and* TDF *be a trapdoor delay function. For every adversary* $\mathcal{A}$ *against* **TPKE-CCA** *there exists adversaries* $\mathcal{B}, \mathcal{C}$ *and* $\mathcal{D}$ *such that*

$$\begin{aligned}\mathsf{Adv}^{(\tau_{\mathsf{pre}},\tau_{\mathsf{post}},\rho_{\mathsf{proc}})\text{-}\mathbf{TPKE\text{-}CCA}}_{\mathsf{NYTPKE}[\mathsf{TDF},\mathsf{NIZK}]}(\mathcal{A}) \leq\ & 2 \cdot \mathsf{Adv}^{(\sigma_{\mathsf{pre}},\sigma_{\mathsf{post}},\pi_{\mathsf{proc}})\text{-}\mathbf{SeqDist}}_{\mathsf{TDF}}(\mathcal{B}) \\ & + \mathsf{Adv}^{\tau_0\text{-}\mathbf{ZK}}_{\mathsf{NIZK}}(\mathcal{C}) + \mathsf{Adv}^{\tau_1\text{-}\mathbf{SS}}_{\mathsf{NIZK}}(\mathcal{D}).\end{aligned}$$

The respective parameters are linked via

$$\tau_0 = (\tau_{\mathsf{pre}} + \tau_{\mathsf{post}}) \cdot t_{\mathsf{vrfy}}, \qquad \tau_1 = \tau_{\mathsf{pre}} + \tau_{\mathsf{post}},$$
$$\sigma_{\mathsf{pre}} = t_{\mathsf{sgen}} + \tau_{\mathsf{pre}} \cdot t_{\mathsf{vrfy}}, \quad \sigma_{\mathsf{post}} = t_{\mathsf{sprv}} + \tau_{\mathsf{post}} \cdot t_{\mathsf{vrfy}}, \quad \pi_{\mathsf{proc}} = \rho_{\mathsf{proc}}.$$

Proof. The proof is almost identical to the proof of [28, Theorem 4]. In the interest of space, the proof is omitted. □

Remark 24. By instantiating NYTPKE with repeated squaring in groups of unknown order as the TDF, we recover the original TPKE construction found in [28].

In [28, Theorem 5] the authors show that there exists a black-box transformation from a TPKE to a NITC.

Theorem 25 (KLX Transformation). *Any TPKE scheme satisfying* **TPKE-CCA** *can be black-box transformed into a NITC scheme satisfying* **NITC-CCA** *and* **NITC-BND**.

Remark 26. The authors of [28] only formally proved that their transformation fulfills **NITC-CCA**. Nevertheless, they informally argue that the obtained NITC scheme also satisfies **NITC-BND**.

Using the KLX transformation we easily transform NYTPKE into a NITC. For completeness, we provide the obtained protocol in Fig. 10. To this end, let NIZK be a NIZK proof system for the relation

$$\mathcal{R} \coloneqq \{(c, (m, r)) \mid c = \mathsf{NYTPKE.Enc}(\mathsf{pk}, m; r)\}$$

and let NIZK′ be a NIZK proof system for the relation

$$\mathcal{R}' \coloneqq \{((c, m), r) \mid c = \mathsf{NYTPKE.Enc}(\mathsf{pk}, m; r)\}.$$

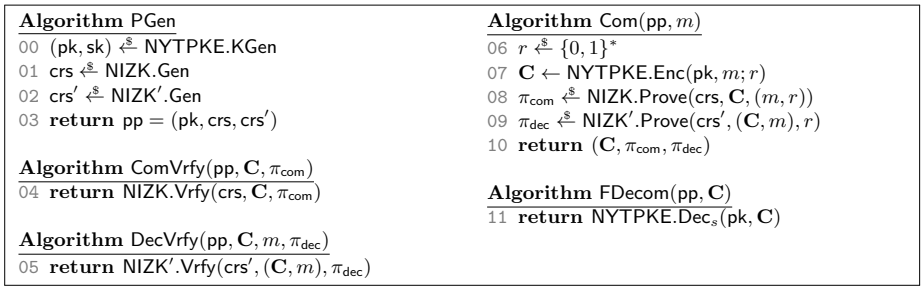

Fig. 10. The KLX transformation applied to NYTPKE.

Corollary 27. *The construction in Fig. 10 satisfies* **NITC-CCA** *and* **NITC-BND**.

5 Post-Quantum Secure Instantiations From Isogenies

In this section, we give instantiations of our constructions from isogenies, leading to one VDF which we call DeuringVDF, and one TDF which we call DeuringTDF. Their names come from the fact that both constructions are based on the isogeny-based verifiable random function DeuringVRF from [30]. We remark that we do not make any *unconditional* statement about the sequentiality of DeuringVDF and DeuringTDF. Instead, we only argue sequentiality when an upper bound on the amount of available parallelism is given. We refer to Section 5.3 for more details.

Throughout this section, p is a prime congruent to 3 modulo 4 and E_0 is a fixed supersingular elliptic curve defined by the equation $y^2 = x^3 + x$. We use $\mathcal{O}_0$ to denote a maximal order in $\mathcal{B}_{p,\infty}$ such that $\mathrm{End}(E_0) \cong \mathcal{O}_0$ via an isomorphism ι_0. We use ζ to denote the automorphism on E_0 that sends (x, y) to $(-x, \alpha y)$ where α is an element in $\mathbb{F}_{p^2}$ such that $\alpha^2 = -1$.

Remark 28. In our constructions, we assume that a curve $E/\mathbb{F}_{p^2}$ is always represented by its unique j-invariant $j(E)$ unless stated otherwise. To avoid convoluted notation, we will not write $j(E)$ explicitly each time.

5.1 DeuringVDF

We now show how to construct a VDF from isogenies using DeuringVRF as basis [30]. DeuringVDF satisfies unconditional uniqueness (Definition 14) and thus allows us to instantiate LEIBNITC.

Let in the following T be a prime such that for $a = \lceil \log T \rceil$, a and T satisfy that $2^a - T = c_1^2 + c_2^2$ for some integers c_1, c_2. Let furthermore $p = f2^eT - 1$ be a prime where $e > a$ is some positive integer and f is a small positive cofactor. We explain how to select all parameters in Section 5.4. We present DeuringVDF in Fig. 11. The input space of DeuringVDF is $\mathcal{X} = \{(E, K) : E \in \mathcal{E}\ell\ell(\mathbb{F}_{p^2}), K \in E[T]\}$ and the output space is $\mathcal{Y} = \{(E, (P, Q)) : E \in \mathcal{E}\ell\ell(\mathbb{F}_{p^2}), (P, Q) \in E[2^a]\}$.

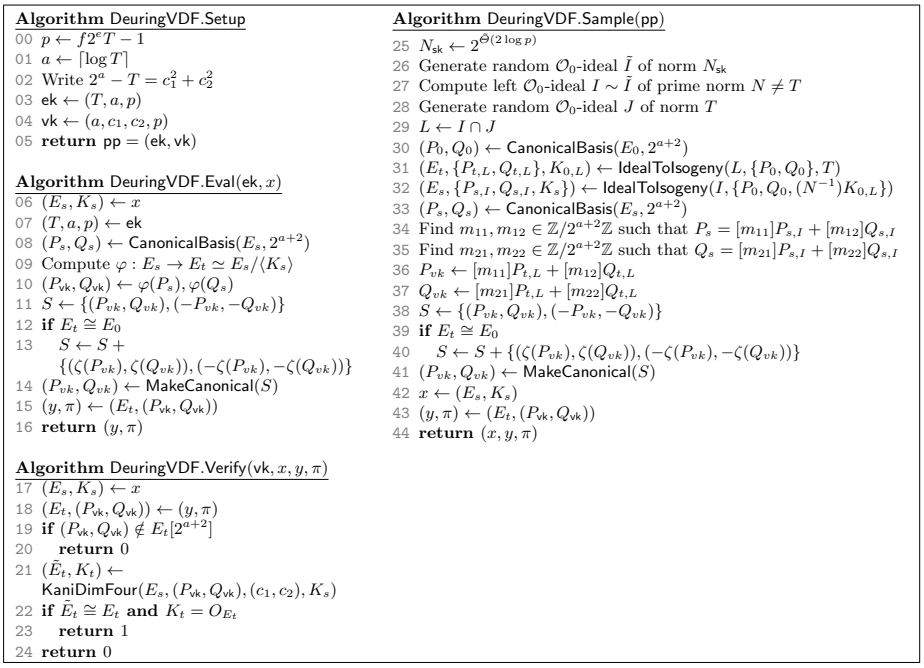

Fig. 11. Our verifiable delay function DeuringVDF.

Theorem 29. DeuringVDF *is correct (Definition 3).*

Proof. Let $\varphi_I, \varphi_J, \varphi_L$ denote the isogenies corresponding to the ideals I, J, L respectively. According to line 31 and 32 in Fig. 11 , the codomain of φ_I is E_s and the codomain of φ_L is E_t. Since $(N, T) = 1$, by design and [22, Lemma 3], $\varphi_L = \varphi'_J \circ \varphi_I$ where $\varphi'_J := [\varphi_I]_* \varphi_J$ is an isogeny from E_s to E_t.

We now show that $\ker(\varphi'_J) = \langle K_s \rangle$, and that $\varphi'_J(P_s, Q_s) = (P_{\mathsf{vk}}, Q_{\mathsf{vk}})$. Since $\varphi_L = \varphi'_J \circ \varphi_I$, this implies that $[N]\widehat{\varphi}'_J = \varphi_I \circ \widehat{\varphi}_L$. Hence $\ker(\varphi'_J) = \widehat{\varphi}'_J(E_t[T]) = \varphi_I \circ [N^{-1}] \circ \widehat{\varphi}_L(E_t[T])$, which is $\langle K_s \rangle$ by line 31 and 32 of Fig. 11. On the other hand

$$\varphi'_J(P_s, Q_s) = \varphi'_J((P_{s,I}, Q_{s,I}) \begin{pmatrix} m_{11} & m_{21} \\ m_{12} & m_{22} \end{pmatrix}) = \varphi'_J \circ \varphi_I((P_0, Q_0) \begin{pmatrix} m_{11} & m_{21} \\ m_{12} & m_{22} \end{pmatrix})$$
$$= \varphi_L((P_0, Q_0) \begin{pmatrix} m_{11} & m_{21} \\ m_{12} & m_{22} \end{pmatrix}) = (P_{t,L}, Q_{t,L}) \begin{pmatrix} m_{11} & m_{21} \\ m_{12} & m_{22} \end{pmatrix},$$

which is $(P_{\mathsf{vk}}, Q_{\mathsf{vk}})$ in Fig. 11.

Given $x = (E_s, K_s), y = E_t, \pi = (P_{\mathsf{vk}}, Q_{\mathsf{vk}})$, and let $y' = E'_t, \pi' = (P'_{\mathsf{vk}}, Q'_{\mathsf{vk}})$ be outputs of DeuringVDF.Eval in Fig. 11 with input $x = (E_s, K_s)$. By design, E'_t is the codomain curve of the isogeny φ from E_s with kernel subgroup $\langle K_s \rangle$, and $(P'_{\mathsf{vk}}, Q'_{\mathsf{vk}})$ is its evaluation at (P_s, Q_s), then it is clear that $(y, \pi) = (y', \pi')$. Note that here the subtlety in proving $\pi = \pi'$ due to the fact that φ_I and φ are

only determined up to automorphisms on E_t is taken care of by the algorithm MakeCanonical.

Furthermore, the evaluation of an isogeny of degree T on 2^{a+2}-torsion of the domain curve E_s determines the isogeny uniquely. Let $\widetilde{E}_t, K_t$ be the output of line 21 of Fig. 11, then by design of KaniDimFour, $\widetilde{E}_t \cong E_t$ and since K_s is the kernel of this isogeny, this implies that $K_t = O_{E_t}$. Hence it holds that $\mathsf{DeuringVDF.Verify}(\mathsf{vk}, x, y, \pi) = 1$. □

Due to Lemma 15 we do not need to prove the soundness of DeuringVDF as it immediately follows from unconditional uniqueness.

Theorem 30. DeuringVDF *is unconditionally unique (Definition 14).*

Proof. Let $(x, y, \pi) \leftarrow \mathsf{DeuringVDF.Sample}(\mathsf{pp})$. Let us write $x = (E_s, K_s)$ and $(y, \pi) = (E_t, (P_{\mathsf{vk}}, Q_{\mathsf{vk}}))$. By the established correctness of DeuringVDF, it follows that $\mathsf{DeuringVDF.Verify}(\mathsf{vk}, x, y, \pi) = 1$. Let (x, y', π') be another triple such that $\mathsf{DeuringVDF.Verify}(\mathsf{vk}, x, y', \pi') = 1$. Let us write $(y', \pi') = (E'_t, (P'_{\mathsf{vk}}, Q'_{\mathsf{vk}}))$, there exists an isogeny of degree T that sends $P_s, Q_s \in E_s[2^{a+2}]$ to $P'_{\mathsf{vk}}, Q'_{\mathsf{vk}}$, and the kernel of this isogeny is $\langle K_s \rangle$. This immediately implies that $j(E_t) = j(E'_t)$ which means $y = y'$, and the fact that $\pi = \pi'$ follows from the discussion in the proof of Theorem 29 regarding the same equality. □

Theorem 31. DeuringVDF *satisfies γ-spreadness (Definition 16) for $\gamma \leq p^{-1/2}$.*

Proof. Let $(E, K) \in \mathcal{X} = \{(E, K) : E \in \mathcal{E}\ell\ell(\mathbb{F}_{p^2}), K \in E[T]\}$ be a random instance. Then one can choose $N_{\mathsf{sk}} = \widetilde{\Theta}(p^2)$ such that

$$\begin{aligned} &\Pr[(E, K) = (E_s, K_s) \mid ((E_s, K_s), \cdot, \cdot) \xleftarrow{\$} \mathsf{DeuringVDF.Sample}(\mathsf{pp})] \\ \leq\ &\Pr[E = E_s \mid ((E_s, \cdot), \cdot, \cdot) \xleftarrow{\$} \mathsf{DeuringVDF.Sample}(\mathsf{pp})] \leq p^{-1/2}. \end{aligned}$$

The last inequality follows from [21, Proposition 29], which measures the statistical distance between the endpoints distribution of random walks on the supersingular 2-isogeny graph and uniform distribution over $\mathcal{E}\ell\ell(\mathbb{F}_{p^2})$. □

The sequentiality of DeuringVDF relies on the following assumption. Its concrete hardness is analyzed in Section 5.3.

Definition 32 (Sequential Isogeny Computability). *Let $p = f2^eT - 1$ be a prime, let $E/\mathbb{F}_{p^2}$ be a random supersingular elliptic curve and let $K \in E[T]$ be a random point of prime order T. Furthermore, let $\mathcal{A}$ be an algorithm with runtime ω_{t} and parallelism ω_{proc}. We define the advantage of $\mathcal{A}$ winning the $(\omega_{\mathsf{t}}, \omega_{\mathsf{proc}})$-**Seq-Isog** game as*

$$\mathsf{Adv}_{p,T}^{(\omega_{\mathsf{t}}, \omega_{\mathsf{proc}})\text{-}\mathbf{Seq\text{-}Isog}}(\mathcal{A}) \coloneqq \Pr[E' = E/\langle K \rangle \mid E' \xleftarrow{\$} \mathcal{A}(E, K)].$$

Algorithm DeuringTDF.Setup

$p \leftarrow f2^eT - 1$
Choose random prime $N_{\mathsf{sk}} \neq p$ of size $\Theta(p)$
Choose random $\mathcal{O}_0$-ideal I of norm N_{sk}
$(P_0, Q_0) \leftarrow E_0[T]$
$(E, P, Q) \leftarrow \mathsf{IdealToIsogeny}(I, \{P_0, Q_0\})$
Choose $\kappa \in \mathrm{End}(E_0)$ with $\kappa(P_0) = Q_0$
$\theta \leftarrow \varphi_I \circ \kappa \circ \hat{\varphi}_I$
$Q \leftarrow [N_{\mathsf{sk}}]Q$
$M = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \xleftarrow{\$} \mathrm{GL}_2(\mathbb{Z}/T\mathbb{Z})$
$(R, S) \leftarrow \big([a]P + [b]Q,\ [c]P + [d]Q\big)$
$\mathsf{ek} \leftarrow (p, T, E, R, S)$
$\mathsf{td} \leftarrow (I, M, \kappa)$
return $(\mathsf{ek}, \mathsf{td})$

Algorithm DeuringTDF.Eval$(\mathsf{ek}, x, \mathsf{td})$

$(p, T, E, R, S) \leftarrow \mathsf{ek}$
$(r, s) \leftarrow f_{\mathrm{in}}(x)$
$P \leftarrow E[2^e]$ \\ Deterministic point
if $\mathsf{td} = \perp$
  $K \leftarrow [r]R + [s]S \in E[T]$
  Compute $\varphi : E \to E_x \simeq E/\langle K \rangle$
  $Q \leftarrow \varphi(P)$
else
  $(I, M, \kappa) \leftarrow \mathsf{td}$
  Parse $M = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \in \mathrm{GL}_2(\mathbb{Z}/T\mathbb{Z})$
  $\gamma \leftarrow (ra + sc) + (rb + sd)\,\kappa$
  Compute I_{P_0} as kernel ideal of $\psi : E_0 \to E_0/\langle P_0 \rangle$
  Compute α with $I_{P_0} = \mathcal{O}_0\langle \alpha, N \rangle$
  $I_x \leftarrow \mathcal{O}_0\langle \alpha\,\overline{\gamma},\ N \rangle$
  $(E_x, Q) \leftarrow \mathsf{IdealToIsogeny}(I \cap I_x, P)$
return (E_x, Q)

Fig. 12. Our trapdoor delay function DeuringTDF.

Theorem 33. *For every adversary $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$ against the $(\sigma_{\mathsf{pre}}, \sigma_{\mathsf{post}}, \pi_{\mathsf{proc}})$-* **SeqVDF** *security of* DeuringVDF *there exists an adversary $\mathcal{B}$ such that*

$$\mathsf{Adv}^{(\sigma_{\mathsf{pre}}, \sigma_{\mathsf{post}}, \pi_{\mathsf{proc}})\text{-}\mathbf{SeqVDF}}_{\mathsf{DeuringVDF}}(\mathcal{A}) \leq \mathsf{Adv}^{(\omega_{\mathsf{t}}, \omega_{\mathsf{proc}})\text{-}\mathbf{Seq\text{-}Isog}}_{p,T}(\mathcal{B}),$$

where $\omega_{\mathsf{proc}} = \pi_{\mathsf{proc}}$ and $\omega_{\mathsf{t}} = \sigma_{\mathsf{post}} + \sigma_{\mathsf{pre}}$.

Proof. On input (p, T, E, K) adversary $\mathcal{B}$ generates $\mathsf{pp} = (\mathsf{ek}, \mathsf{vk})$ based on p and T as in Fig. 11 and forwards pp to $\mathcal{A}_0$. When $\mathcal{A}_0$ outputs a state L, adversary $\mathcal{B}$ simply forwards its own challenge $x = (E, K)$ as well was the state L to $\mathcal{A}_1$. Note that the tuple (E, K) is a correctly distributed DeuringVDF instance. Indeed, $N_{\mathsf{sk}} \geq 2 \log p$ and therefore the distribution of curves output by VDF.Sample is statistically close to the uniform distribution [21, Proposition 29]. When $\mathcal{A}_1$ outputs a solution $y = E_t$ as well as a proof $\pi = (P_{\mathsf{vk}}, Q_{\mathsf{vk}})$, $\mathcal{B}$ can simply forward E_t to the **Seq-Isog** game. □

5.2 DeuringTDF

DeuringTDF is conceptually very similar to DeuringVRF with slight modifications in parameter choices and algorithms. We include the modified algorithms in Fig. 12. Let in the following $p = f2^eT - 1$ be a prime and $f_{\mathrm{in}} : \{0,1\}^\delta \to \mathbb{P}^1(\mathbb{Z}/T\mathbb{Z})$ be an injective function for $\delta = \lfloor \log T \rfloor$. Subsequently, the input space of DeuringTDF is $\mathcal{X} = \{0,1\}^\delta$ and the output space is $\mathcal{Y} = \{(E, Q) : E \in \mathcal{E}\ell\ell(\mathbb{F}_{p^2}), Q \in E[2^e]\}$. Parameter selection is again deferred to Section 5.4.

Theorem 34. DeuringTDF *satisfies correctness (Definition 21).*

Proof. Let φ_I, φ_{I_x} denote the isogenies corresponding to the ideals I, I_x respectively. First notice that the ideal I_x is the kernel ideal of the point $[r]([a]P_0 +$

$[b]Q_0) + [s]([c]P_0 + [d]Q_0) \in E_0[T]$ (see the proof of [25, Proposition 1]). This shows that

$$\begin{aligned}\ker([\varphi_I]_*(\varphi_{I_x})) = \varphi_I(\ker \varphi_{I_x}) &= \varphi_I([r]([a]P_0 + [b]Q_0) + [s]([c]P_0 + [d]Q_0)) \\ &= [r]R + [s]S.\end{aligned}$$

Since $\mathsf{IdealToIsogeny}(I \cap I_x, P)$ computes the codomain of the isogeny $[\varphi_I]_*(\varphi_{I_x}) \circ \varphi_I$ and its evaluation on $P \in E[2^e]$, we have that (E_x, Q) computed by the algorithm $\mathsf{IdealToIsogeny}(I \cap I_x, P)$ is exactly the codomain curve of the isogeny from E with kernel $[r]R + [s]S$ and its evaluation on P. □

The sequentiality of DeuringTDF relies on the following decisional assumption. Its concrete hardness is analyzed in Section 5.3.

Definition 35 (Sequential Isogeny Distinguishability). *Let $p = f2^eT - 1$ be a prime, let $E/\mathbb{F}_{p^2}$ be a random supersingular elliptic curve and let $K \in E[T]$ be a random point of prime order T. Furthermore, let $\varphi : E \to E_1$ be the isogeny with $\ker \varphi = \langle K \rangle$ and let $Q_1 = \varphi(P_1)$ for a deterministically chosen $P_1 \in E[2^e]$. Let $E_2/\mathbb{F}_{p^2}$ be another random curve and $Q_2 \in E_2[2^e]$ be a random point. Lastly, let $\mathcal{A}$ be an algorithm runtime ω_t and parallelism ω_proc. We define the advantage of $\mathcal{A}$ winning the $(\omega_\mathsf{t}, \omega_\mathsf{proc})$-**Seq-Isog-Dist** game as*

$$\mathsf{Adv}^{(\omega_\mathsf{t},\omega_\mathsf{proc})\text{-}\mathbf{Seq\text{-}Isog\text{-}Dist}}_{p,T}(\mathcal{A}) := \left| \Pr\left[b = b' \;\middle|\; \begin{matrix} b \xleftarrow{\$} \{1,2\} \\ b' \xleftarrow{\$} \mathcal{A}(E, K, E_b, Q_b) \end{matrix} \right] - \frac{1}{2} \right|.$$

Theorem 36. *For every adversary $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$ against the $(\sigma_\mathsf{pre}, \sigma_\mathsf{post}, \pi_\mathsf{proc})$-**SeqDist** security of DeuringTDF there exists an adversary $\mathcal{B}$ such that*

$$\mathsf{Adv}^{(\sigma_\mathsf{pre},\sigma_\mathsf{post},\pi_\mathsf{proc})\text{-}\mathbf{SeqDist}}_{\mathsf{DeuringTDF}}(\mathcal{A}) \leq \mathsf{Adv}^{(\omega_\mathsf{t},\omega_\mathsf{proc})\text{-}\mathbf{Seq\text{-}Isog\text{-}Dist}}_{p,T}(\mathcal{B}),$$

where $\omega_\mathsf{proc} = \pi_\mathsf{proc}$ and $\omega_\mathsf{t} = \sigma_\mathsf{post} + \sigma_\mathsf{pre}$.

Proof. The proof is almost identical to Theorem 33 and therefore omitted. □

5.3 Parallelization of Isogeny Computations

We now analyze the sequentiality of **Seq-Isog** and **Seq-Isog-Dist** (Definitions 32 and 35). To this end, we argue that the best strategy to win either game is to compute the prime-degree isogeny $\varphi : E \to E/\langle K \rangle$ directly from its kernel. We discuss alternative approaches that are all outperformed by parallel variants of Vélu and $\sqrt{\text{élu}}$.

The Vélu Family. The baseline for our analysis are the Vélu and $\sqrt{\text{élu}}$ algorithms [8, 37]. As mentioned before, Vélu can be parallelized perfectly via a binary tree strategy. Notably, the runtime of parallel Vélu only depends on $T = \text{ord}(K)$ and the overhead induced by combining the results from the parallel threads is almost negligible.

Proposition 37. *Let p be a prime, let $E/\mathbb{F}_{p^2}$ be a supersingular elliptic curve and $K \in E[T]$ be an accessible point of prime order T. The parallel cost of computing $\varphi : E \to E/\langle K \rangle$ via Vélu's formula is*

$$C_{\mathsf{par}}^{\mathsf{Vel}} = T/n + \log n \tag{1}$$

field operations with n the amount of parallelism.

The sequential as well as the parallel performance of $\sqrt{\text{élu}}$ were analyzed in [19] and we recall the relevant results below. As with plain Vélu, the runtime only depends T and n.

Proposition 38. *Let p be a prime, let $E/\mathbb{F}_{p^2}$ be a supersingular elliptic curve and $K \in E[T]$ be an accessible point of prime order T. The sequential cost of computing $\varphi : E \to E/\langle K \rangle$ is*

$$C_{\mathsf{seq}}^{\mathsf{sqrtVel}} = 32b^{\log 3} + 8\log b + 16b + 12$$

field operations with $b = \sqrt{T-1}/2$. Likewise, the parallel cost of computing $\varphi : E \to E/\langle K \rangle$ is

$$C_{\mathsf{par}}^{\mathsf{sqrtVel}} = \frac{62}{3}b^{\log 3} + 6(n^2 - n + 2)\left(\frac{b}{n}\right)^{\log 3} + 7\log b + \frac{13b}{n} + 16b + \frac{3}{2}\log n + \frac{39}{2}n - \frac{25}{2} \tag{2}$$

field operations with n the amount of parallelism and $b = \sqrt{(T-1)/2n}$.

As evident by Eq. (2) the overhead of combining results from different threads of the computation is not negligible for parallel $\sqrt{\text{élu}}$ and becomes dominant once too much parallelism is used. Indeed, ignoring sub-dominant terms as well as approximating $b^{\log 3} \approx b$ and $n^{-\log 3} \approx n^{-2}$, we have that Eq. (2) scales roughly as $\frac{128}{3}\sqrt{(T-1)/2n} + \frac{39n}{2}$. Note that this yields a lower bound since the former term appears exclusively in the numerator while the latter term appears exclusively in the denominator. Therefore, $\sqrt{\text{élu}}$ cannot be parallelized perfectly and instead there is an *optimal* amount of parallelization, after which more parallelization is not helpful anymore.

Remark 39. Both cost functions in Proposition 38 include the cost of evaluating φ on two torsion points as well as computing the codomain.

Recovering End(E). Alternatively, an adversary could also compute $\mathrm{End}(E)$ and recover $E/\langle K \rangle$ from the kernel ideal associated to the isogeny determined by $\langle K \rangle$. However, the computational complexity recovering $\mathrm{End}(E)$ is well understood (see for instance [1, 21]) and the strategy is easily prevented by choosing a prime characteristic of appropriate size. Indeed, computing $\mathrm{End}(E)$ depends exclusively on the field size and is independent on $T = \mathrm{ord}(K)$. Currently, the fastest algorithms compute $\mathrm{End}(E)$ in time $\mathcal{O}(p^{1/2})$.

Modular Polynomials. The classical T-modular polynomial $\Phi_T(X, Y)$ is a bivariate polynomial in $\mathbb{Z}[X, Y]$ that encodes the j-invariants of all pairs of T-isogenous curves. As a result, a valid strategy of computing the codomain $E/\langle K\rangle$ is to substitute $j(E)$ into $\Phi(X, Y)$ and subsequently compute all roots of the resulting univariate polynomial $\Phi(j(E), Y)$ via a parallelized Chinese remainder algorithm as suggested by Sutherland [35]. However, this approach does not allow for the *evaluation* of the isogeny $\varphi : E \to E/\langle K\rangle$ on torsion points, which is necessary for both DeuringVDF and DeuringTDF. Therefore, an adversary would need to guess at least one image $\varphi(P)$ of a point $P \in E[2^e]$ of large order.[9] Since $\#E[2^e] = 2^{2e}$, we conclude that this strategy has complexity $\mathcal{O}(2^{2e})$.

5.4 Parameters and Compactness

Both DeuringVDF and DeuringTDF require a prime of the form $p = f2^eT - 1$ with T a prime determining the desired delay. Furthermore, there are several other requirements to consider that govern the security of both protocols.

Computing T. We want to select a prime T such that

$$\min\left\{C_{\mathsf{par}}^{\mathsf{Vel}},\, C_{\mathsf{par}}^{\mathsf{sqrtVel}}\right\} \geq t$$

for a fixed choice of t and n. To this end, we substitute n and t into Eqs. (1) and (2) and solve the equations for T. Afterwards, we choose the larger of the two results as our delay parameter T.

The Prime Characteristic. For DeuringVDF and DeuringTDF we select a prime $p = f2^eT - 1$ in the following way. For fixed parameters t and n we first compute T as described above. Subsequently, we set $a = \lceil \log T \rceil$ and choose $e = \max\{4\lambda - (a + 2), a + 2\}$, guaranteeing that $\log p \geq 4\lambda$, that $E[2^{a+2}]$ is accessible and that $2^e > \sqrt{p}$ (which is a requirement for IdealToIsogeny). The requirement $\log p \geq 4\lambda$ is necessary to ensure a spreadness $\gamma \leq 2^{-2\lambda}$, resulting in tight security reductions. Furthermore, this allows us to argue that computing $\mathrm{End}(E)$ is much harder than computing an isogeny from its kernel. Furthermore, we require that $2^a - T$ is a sum of two squares to make the 4-dimensional representation of the T-isogeny φ in DeuringVDF possible.[10] If this is not the case, we set T to be the next largest prime and repeat the previous steps. Lastly, we choose a small cofactor f to ensure primality of p, resulting in $p \approx \max\{2^{4\lambda}, T^2\}$ depending on the delay t. Concretely, for $\lambda = 128$, $t = 2^{32}$ and $n = 2^{16}$ we propose the 518-bit prime

$$p_{\{128,32,16\}} = 3 \cdot 5^2 \cdot 2^{463} \cdot \underbrace{281474976710899}_{T} - 1.$$

Primes for different choices of λ, t and n can easily be generated, even for large values of t and n.

[9] Note that in DeuringVDF the other torsion point can be recovered from a pairing computation.

[10] This requirement is optional for DeuringTDF.

Compactness of LEIBNITC. Instantiating LEIBNITC with DeuringVDF yields a very compact scheme. Regarding the public parameter $\mathsf{pp} = (\mathsf{ek}, \mathsf{vk})$, the evaluation key $\mathsf{ek} = (T, a, p)$ consist only of integers and has size at most $2\log p$ since $\log T \leq \log(p)/2$ and $\log a \leq \log\log p$. Likewise, the verification key $\mathsf{vk} = (a, c_1, c_2, p)$ also consist of integers and has size at most $3\log p$. The commitment $\mathbf{C} = (E_s, K_s, u)$ consist of a curve $E_s/\mathbb{F}_{p^2}$, a point $K_s \in E_s[T]$ and a symmetric ciphertext u of a message m, requiring at most $2\log p + 2\log p + 256$ bits assuming $|m| = 256$ bits. The decommitment proof $\pi_{\mathsf{dec}} = (E_t, P_{\mathsf{vk}}, Q_{\mathsf{vk}})$ consists of a curve $E_t/\mathbb{F}_{p^2}$ and two torsion points $(P_{\mathsf{vk}}, Q_{\mathsf{vk}}) \in E_t[2^{a+2}]$, requiring $2\log p + 3a + 6$ bits using point compression. Lastly, $\pi_{\mathsf{com}} = \perp$ requires no space. Using the concrete prime $p_{128,32,16}$ proposed above, we get that $|\mathsf{pp}| = 1141$, $|\mathbf{C}| = 2328$ and $|\pi_{\mathsf{dec}}| = 1189$ bits. In comparison, the NITC in [17] has $|\mathsf{pp}| = 15360$, $|\mathbf{C}| = 12320$ and $|\pi_{\mathsf{dec}}| = 12400$ bits using the same choice for λ, t and $|m|$.

Compactness of NYTPKE. The concrete size of NYTPKE instantiated with DeuringTDF is difficult to estimate due to its reliance on a NIZK proof system. In general, we have $\mathsf{pk} = (\mathsf{ek}_1, \mathsf{ek}_2, \mathsf{crs})$ and $\mathsf{sk} = (\mathsf{crs}, \mathsf{td}_1)$. The evaluation keys $\mathsf{ek}_i = (p_i, T_i, E_i, R_i, S_i)$ consist of two primes p_i and T_i, a curve $E_i/\mathbb{F}_{p^2}$ and two points $(R_i, S_i) \in E_i[T]$ and thus have size $|\mathsf{ek}_i| = 15\log p/2$ each since $\log(T_i) \leq \log(p)/2$ and $|E_i| = |R_i| = |S_i| = 2\log p$. The trapdoor $\mathsf{td}_1 = (I, M, \kappa)$ consists of an ideal I, a matrix $M \in \mathrm{GL}_2(\mathbb{Z}/T\mathbb{Z})$ and an endomorphism κ. The ideal I can be represented with $5\log p$ bits by writing it in terms of an $\mathcal{O}_0$-basis, the matrix M requires $2\log p$ bits (note that $T \leq \sqrt{p}$) and the endomorphism κ requires $4\log p$ bits, yielding $|\mathsf{td}_1| = 11\log p$. A ciphertext $\mathsf{ct} = (x_1, x_2, z_1, z_2, \pi)$ consist of two DeuringTDF instances (x_1, x_2), two messages (z_1, z_2) and a NIZK proof π. We have $|x_i| = 2\log T \leq \log p$ and $|z_i| = |m|$. In total, we get $|\mathsf{pk}| = |\mathsf{crs}| + 15\log p$, $|\mathsf{sk}| = |\mathsf{crs}| + 11\log p$ and $|\mathsf{ct}| = |\pi| + 2|m| + 4\log p$.

Acknowledgments. We would like to thank Jonas Janneck for his helpful discussions and feedback on the security proofs. Jonas Meers was funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy – EXC 2092 CASA – 390781972. Mingjie Chen is supported partly by Fonds voor Wetenschappelijk Onderzoek (FWO) with reference number 12A4L26N, partly by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement ISOCRYPT – No. 101020788), and partly by the Research Council KU Leuven grant C14/24/099 and by CyberSecurity Research Flanders with reference number VOEWICS02.

References

1. Aardal, M.A., Adj, G., Aranha, D.F., Basso, A., Canales Martínez, I.A., Chávez-Saab, J., Santos, M.C., Dartois, P., De Feo, L., Duparc, M., Eriksen, J.K.,

Fouotsa, T.B., Filho, D.L.G., Hess, B., Kohel, D., Leroux, A., Longa, P., Maino, L., Meyer, M., Nakagawa, K., Onuki, H., Panny, L., Patranabis, S., Petit, C., Pope, G., Reijnders, K., Robert, D., Rodríguez Henríquez, F., Schaeffler, S., Wesolowski, B.: SQIsign. Tech. rep., National Institute of Standards and Technology (2024), available at https://csrc.nist.gov/Projects/pqc-dig-sig/round-2-additional-signatures
2. Ahrens, K.: SIGNITC: Supersingular isogeny graph non-interactive timed commitments. Cryptology ePrint Archive, Report 2024/1225 (2024), https://eprint.iacr.org/2024/1225
3. Ambrona, M., Beunardeau, M., Toledo, R.R.: Timed commitments revisited. Cryptology ePrint Archive, Report 2023/977 (2023), https://eprint.iacr.org/2023/977
4. Basso, A., Borin, G., Castryck, W., Santos, M.C.R., Invernizzi, R., Leroux, A., Maino, L., Vercauteren, F., Wesolowski, B.: PRISM: Simple and compact identification and signatures from large prime degree isogenies. In: Jager, T., Pan, J. (eds.) PKC 2025, Part III. LNCS, vol. 15676, pp. 300–332. Springer, Cham (May 2025). https://doi.org/10.1007/978-3-031-91826-1_10
5. Basso, A., Dartois, P., De Feo, L., Leroux, A., Maino, L., Pope, G., Robert, D., Wesolowski, B.: SQIsign2D-West - the fast, the small, and the safer. In: Chung, K.M., Sasaki, Y. (eds.) ASIACRYPT 2024, Part III. LNCS, vol. 15486, pp. 339–370. Springer, Singapore (Dec 2024). https://doi.org/10.1007/978-981-96-0891-1_11
6. Basso, A., Maino, L.: POKÉ: A compact and efficient PKE from higher-dimensional isogenies. In: Fehr, S., Fouque, P.A. (eds.) EUROCRYPT 2025, Part II. LNCS, vol. 15602, pp. 94–123. Springer, Cham (May 2025). https://doi.org/10.1007/978-3-031-91124-8_4
7. Baum, C., David, B., Dowsley, R., Nielsen, J.B., Oechsner, S.: TARDIS: A foundation of time-lock puzzles in UC. In: Canteaut, A., Standaert, F.X. (eds.) EUROCRYPT 2021, Part III. LNCS, vol. 12698, pp. 429–459. Springer, Cham (Oct 2021). https://doi.org/10.1007/978-3-030-77883-5_15
8. Bernstein, D.J., De Feo, L., Leroux, A., Smith, B.: Faster computation of isogenies of large prime degree. Open Book Series **4**(1), 39–55 (Dec 2020). https://doi.org/10.2140/obs.2020.4.39, http://dx.doi.org/10.2140/obs.2020.4.39
9. Biasse, J.F., Song, F.: Efficient quantum algorithms for computing class groups and solving the principal ideal problem in arbitrary degree number fields. In: Proceedings of the twenty-seventh annual ACM-SIAM symposium on Discrete algorithms. pp. 893–902. SIAM (2016)
10. Bitansky, N., Goldwasser, S., Jain, A., Paneth, O., Vaikuntanathan, V., Waters, B.: Time-lock puzzles from randomized encodings. In: Sudan, M. (ed.) ITCS 2016. pp. 345–356. ACM (Jan 2016). https://doi.org/10.1145/2840728.2840745
11. Boneh, D., Bonneau, J., Bünz, B., Fisch, B.: Verifiable delay functions. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018, Part I. LNCS, vol. 10991, pp. 757–788. Springer, Cham (Aug 2018). https://doi.org/10.1007/978-3-319-96884-1_25
12. Boneh, D., Halevi, S., Hamburg, M., Ostrovsky, R.: Circular-secure encryption from decision Diffie-Hellman. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 108–125. Springer, Berlin, Heidelberg (Aug 2008). https://doi.org/10.1007/978-3-540-85174-5_7
13. Boneh, D., Naor, M.: Timed commitments. In: Bellare, M. (ed.) CRYPTO 2000. LNCS, vol. 1880, pp. 236–254. Springer, Berlin, Heidelberg (Aug 2000). https://doi.org/10.1007/3-540-44598-6_15

14. Borin, G., Santos, M.C.R., Eriksen, J.K., Invernizzi, R., Mula, M., Schaeffler, S., Vercauteren, F.: Qlapoti: Simple and efficient translation of quaternion ideals to isogenies. Cryptology ePrint Archive, Paper 2025/1604 (2025), https://eprint.iacr.org/2025/1604
15. Brakerski, Z., Döttling, N., Garg, S., Malavolta, G.: Leveraging linear decryption: Rate-1 fully-homomorphic encryption and time-lock puzzles. In: Hofheinz, D., Rosen, A. (eds.) TCC 2019, Part II. LNCS, vol. 11892, pp. 407–437. Springer, Cham (Dec 2019). https://doi.org/10.1007/978-3-030-36033-7_16
16. Chávez-Saab, J., Ortega, O., Pizarro-Madariaga, A.: On the parallelization of square-root Vélu's formulas. Cryptology ePrint Archive, Report 2024/851 (2024), https://eprint.iacr.org/2024/851
17. Chvojka, P., Jager, T.: Simple, fast, efficient, and tightly-secure non-malleable non-interactive timed commitments. In: Boldyreva, A., Kolesnikov, V. (eds.) PKC 2023, Part I. LNCS, vol. 13940, pp. 500–529. Springer, Cham (May 2023). https://doi.org/10.1007/978-3-031-31368-4_18
18. Chvojka, P., Jager, T., Slamanig, D., Striecks, C.: Versatile and sustainable timed-release encryption and sequential time-lock puzzles (extended abstract). In: Bertino, E., Shulman, H., Waidner, M. (eds.) ESORICS 2021, Part II. LNCS, vol. 12973, pp. 64–85. Springer, Cham (Oct 2021). https://doi.org/10.1007/978-3-030-88428-4_4
19. Chávez-Saab, J., Ortega, O., Pizarro-Madariaga, A.: On the parallelization of square-root vélu's formulas. Mathematical and Computational Applications **29**(1) (2024). https://doi.org/10.3390/mca29010014, https://www.mdpi.com/2297-8747/29/1/14
20. Dartois, P.: Fast computation of 2-isogenies in dimension 4 and cryptographic applications. Journal of Algebra **683**, 449–514 (2025). https://doi.org/https://doi.org/10.1016/j.jalgebra.2025.06.033, https://www.sciencedirect.com/science/article/pii/S0021869325003771
21. Dartois, P., Leroux, A., Robert, D., Wesolowski, B.: SQIsignHD: New dimensions in cryptography. In: Joye, M., Leander, G. (eds.) EUROCRYPT 2024, Part I. LNCS, vol. 14651, pp. 3–32. Springer, Cham (May 2024). https://doi.org/10.1007/978-3-031-58716-0_1
22. De Feo, L., Kohel, D., Leroux, A., Petit, C., Wesolowski, B.: SQISign: Compact post-quantum signatures from quaternions and isogenies. In: Moriai, S., Wang, H. (eds.) ASIACRYPT 2020, Part I. LNCS, vol. 12491, pp. 64–93. Springer, Cham (Dec 2020). https://doi.org/10.1007/978-3-030-64837-4_3
23. De Santis, A., Di Crescenzo, G., Ostrovsky, R., Persiano, G., Sahai, A.: Robust non-interactive zero knowledge. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 566–598. Springer, Berlin, Heidelberg (Aug 2001). https://doi.org/10.1007/3-540-44647-8_33
24. Decru, T., Maino, L., Sanso, A.: Towards a quantum-resistant weak verifiable delay function. In: Aly, A., Tibouchi, M. (eds.) LATINCRYPT 2023. LNCS, vol. 14168, pp. 149–168. Springer, Cham (Oct 2023). https://doi.org/10.1007/978-3-031-44469-2_8
25. Deuring, M.: Die typen der multiplikatorenringe elliptischer funktionenkörper. Abhandlungen aus dem Mathematischen Seminar der Universität Hamburg **14**(1), 197–272 (Dec 1941). https://doi.org/10.1007/BF02940746, https://doi.org/10.1007/BF02940746
26. Freitag, C., Komargodski, I., Pass, R., Sirkin, N.: Non-malleable time-lock puzzles and applications. In: Nissim, K., Waters, B. (eds.) TCC 2021, Part III. LNCS,

vol. 13044, pp. 447–479. Springer, Cham (Nov 2021). https://doi.org/10.1007/978-3-030-90456-2_15
27. Jao, D., De Feo, L.: Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. In: Yang, B.Y. (ed.) Post-Quantum Cryptography - 4th International Workshop, PQCrypto 2011. pp. 19–34. Springer, Berlin, Heidelberg (Nov / Dec 2011). https://doi.org/10.1007/978-3-642-25405-5_2
28. Katz, J., Loss, J., Xu, J.: On the security of time-lock puzzles and timed commitments. In: Pass, R., Pietrzak, K. (eds.) TCC 2020, Part III. LNCS, vol. 12552, pp. 390–413. Springer, Cham (Nov 2020). https://doi.org/10.1007/978-3-030-64381-2_14
29. Kohel, D., Lauter, K., Petit, C., Tignol, J.P.: On the quaternion ℓ-isogeny path problem. LMS Journal of Computation and Mathematics **17**(A), 418–432 (2014)
30. Leroux, A.: Verifiable random function from the deuring correspondence and higher dimensional isogenies. In: Fehr, S., Fouque, P.A. (eds.) EUROCRYPT 2025, Part VII. LNCS, vol. 15607, pp. 167–194. Springer, Cham (May 2025). https://doi.org/10.1007/978-3-031-91098-2_7
31. Malavolta, G., Thyagarajan, S.A.K.: Homomorphic time-lock puzzles and applications. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019, Part I. LNCS, vol. 11692, pp. 620–649. Springer, Cham (Aug 2019). https://doi.org/10.1007/978-3-030-26948-7_22
32. Naor, M., Yung, M.: Public-key cryptosystems provably secure against chosen ciphertext attacks. In: 22nd ACM STOC. pp. 427–437. ACM Press (May 1990). https://doi.org/10.1145/100216.100273
33. Onuki, H., Nakagawa, K.: Ideal-to-isogeny algorithm using 2-dimensional isogenies and its application to SQIsign. In: Chung, K.M., Sasaki, Y. (eds.) ASIACRYPT 2024, Part III. LNCS, vol. 15486, pp. 243–271. Springer, Singapore (Dec 2024). https://doi.org/10.1007/978-981-96-0891-1_8
34. Rivest, R.L., Shamir, A., Wagner, D.A.: Time-lock puzzles and timed-release crypto. Tech. rep., USA (1996)
35. Sutherland, A.: On the evaluation of modular polynomials. Open Book Ser. **1**(1), 531–555 (Nov 2013)
36. Thyagarajan, S.A.K., Castagnos, G., Laguillaumie, F., Malavolta, G.: Efficient CCA timed commitments in class groups. In: Vigna, G., Shi, E. (eds.) ACM CCS 2021. pp. 2663–2684. ACM Press (Nov 2021). https://doi.org/10.1145/3460120.3484773
37. Vélu, J.: Isogénies entre courbes elliptiques. Comptes-Rendus de l'Académie des Sciences **273**, 238–241 (1971)

Supplementary Material

A Further Preliminaries on Simulation-Sound NIZK

We recall the definition of a simulation-sound non-interactive zero-knowledge (NIZK) proof system [23, 28]. Let for the remainder of the paper $\mathcal{L}_{\mathcal{R}}$ be an NP language defined by a relation $\mathcal{R}$.

Definition 40 (NIZK Proof System). *A* $(t_{\mathsf{prv}}, t_{\mathsf{vrfy}}, t_{\mathsf{sgen}}, t_{\mathsf{sprv}})$-Non-Interactive Zero-Knowledge Proof System *is a tuple of algorithms* $\mathsf{NIZK} = (\mathsf{NIZK.Gen}, \mathsf{NIZK.Prove}, \mathsf{NIZK.Vrfy}, \mathsf{NIZK.SimGen}, \mathsf{NIZK.SimProve})$ *with the following behavior:*

- $\mathsf{crs} \xleftarrow{\$} \mathsf{NIZK.Gen}$*: The randomized parameter generation algorithm outputs a common reference string* crs*.*
- $\pi \xleftarrow{\$} \mathsf{NIZK.Prove}(\mathsf{crs}, x, w)$*: The randomized prover algorithm takes as input a common reference string* crs *as well as* $(x, w) \in \mathcal{R}$ *and outputs a proof* π *in time at most* t_{prv}*.*
- $\{0,1\} \leftarrow \mathsf{NIZK.Vrfy}(\mathsf{crs}, x, \pi)$*: The deterministic verification algorithm takes as input a common reference string* crs*, a statement* x *and a proof* π *and outputs a bit in time* t_{vrfy}*.*
- $(\mathsf{crs}, \mathsf{td}_{\mathsf{zk}}) \xleftarrow{\$} \mathsf{NIZK.SimGen}$*: The randomized simulated parameter generation algorithm outputs a common reference string* crs *and a trapdoor* $\mathsf{td}_{\mathsf{zk}}$ *in time* t_{sgen}*.*
- $\pi \xleftarrow{\$} \mathsf{NIZK.SimProve}(x, \mathsf{td}_{\mathsf{zk}})$*: The randomized simulated prover algorithm takes as input a statement* x *and a trapdoor* $\mathsf{td}_{\mathsf{zk}}$ *and outputs a proof* π *in time* t_{sprv}*.*

We require perfect completeness*: For all* $\mathsf{crs} \xleftarrow{\$} \mathsf{NIZK.Gen}$*, all* $(x, w) \in \mathcal{R}$ *and all* $\pi \xleftarrow{\$} \mathsf{NIZK.Prove}(\mathsf{crs}, x, w)$ *we have* $\mathsf{NIZK.Vrfy}(\mathsf{crs}, x, \pi) = 1$*.*

Definition 41 (Soundness). *A NIZK proof system* NIZK *is* (ϵ, τ)-sound *if for all algorithms* $\mathcal{A}$ *that run in time* τ

$$\Pr\left[\begin{array}{c}\mathsf{NIZK.Vrfy}(\mathsf{crs}, x, \pi) = 1 \\ x \notin \mathcal{R}\end{array}\middle|\begin{array}{c}\mathsf{crs} \xleftarrow{\$} \mathsf{NIZK.Gen} \\ (x, \pi) \leftarrow \mathcal{A}(\mathsf{crs})\end{array}\right] \leq \epsilon.$$

Definition 42 (Zero-Knowledge). *Let* NIZK *be a NIZK proof system and let* $\mathcal{A}$ *be an algorithm with total runtime* τ*. Consider the game* τ-**ZK** *in Fig. 13. We define the advantage of* $\mathcal{A}$ *winning the* τ-**ZK** *game as*

$$\mathsf{Adv}^{\tau\text{-}\mathbf{ZK}}_{\mathsf{NIZK}} := \left|\Pr[\tau\text{-}\mathbf{ZK}_{\mathsf{NIZK}}(\mathcal{A}) \Rightarrow 1] - \frac{1}{2}\right|.$$

Definition 43 (Simulation Soundness). *Let* NIZK *be a NIZK proof system and let* $\mathcal{A}$ *be an algorithm with total runtime* τ*. Consider the game* τ-**SS** *in Fig. 14. We define the advantage of* $\mathcal{A}$ *winning the* τ-**SS** *game as*

$$\mathsf{Adv}^{\tau\text{-}\mathbf{SS}}_{\mathsf{NIZK}} := \Pr[\tau\text{-}\mathbf{SS}_{\mathsf{NIZK}}(\mathcal{A}) \Rightarrow 1].$$

Game τ-$\mathbf{ZK}_{\mathsf{NIZK}}(\mathcal{A})$	**Oracle** $\mathsf{Prove}(x, w)$
00 $b \xleftarrow{\$} \{0,1\}$	05 **if** $(x, w) \notin \mathcal{R}$
01 $\mathsf{crs}_0 \xleftarrow{\$} \mathsf{NIZK.Gen}$	06 **return** $\perp$
02 $(\mathsf{crs}_1, \mathsf{td}_{\mathsf{zk}}) \xleftarrow{\$} \mathsf{NIZK.SimGen}$	07 $\pi_0 \xleftarrow{\$} \mathsf{NIZK.Prove}(\mathsf{crs}_0, x, w)$
03 $b' \leftarrow \mathcal{A}^{\mathsf{Prove}}(\mathsf{crs}_b)$	08 $\pi_1 \xleftarrow{\$} \mathsf{NIZK.SimProve}(x, \mathsf{td}_{\mathsf{zk}})$
04 **return** $[\![b = b']\!]$	09 **return** π_b

Fig. 13. Game defining τ-**ZK** for a NIZK proof system NIZK.

Game τ-$\mathbf{SS}_{\mathsf{NIZK}}(\mathcal{A})$	**Oracle** $\mathsf{SProve}(x)$
00 $(\mathsf{crs}, \mathsf{td}_{\mathsf{zk}}) \xleftarrow{\$} \mathsf{NIZK.SimGen}$	04 $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{x\}$
01 $\mathcal{Q} \leftarrow \emptyset$	05 $\pi \xleftarrow{\$} \mathsf{NIZK.SimProve}(x, \mathsf{td}_{\mathsf{zk}})$
02 $(x, \pi) \leftarrow \mathcal{A}^{\mathsf{SProve}}(\mathsf{crs})$	06 **return** π
03 **return** $[\![\mathsf{NIZK.Vrfy}(\mathsf{crs}, x, \pi) = 1 \wedge x \notin \mathcal{L}_{\mathcal{R}} \wedge x \notin \mathcal{Q}]\!]$	

Fig. 14. Game defining τ-**SS** for a NIZK proof system NIZK.

B On the Ideal to Isogeny Algorithm

Let $p = f2^eT - 1$ be a prime such that $2^e \geq \sqrt{p}$ as in our parameter choice for DeuringVDF. In this section, we explain the details of the ideal to isogeny algorithm $\mathsf{IdealToIsogeny}(I, S, N)$. Specifically, given a left $\mathcal{O}_0$-ideal I such that $(n(I), 2) = 1$, a set S consisting of points on E_0, and an odd prime N such that $N \mid n(I)$, the goal is to compute the codomain E of the associated isogeny φ_I, the evaluation of φ_I on points in S and a generator of $\hat{\varphi}_I(E[N])$. In practice we only take S to be a basis of $E_0[2^e]$, and we work under this assumption in what follows.

For this, we adapt the ideal to isogeny algorithm $\mathsf{IdealToIsogenyIQO}$ from [33, Algorithm 3]. We introduce $\mathsf{GenIdealToIsogenyIQO}$ which is obtained by modifying $\mathsf{IdealToIsogenyIQO}$ slightly and we briefly discuss how the algorithm works in Remark 44. In Algorithm 1, we will call $\mathsf{GenIdealToIsogenyIQO}$ on inputs $\mathcal{O}_0, I', S_{odd}$ where $\mathcal{O}_0$ should be thought of as the trivial left $\mathcal{O}_0$-ideal of norm 1, I' is a left $\mathcal{O}_0$-ideal equivalent to I such that $n(I')$ is a power of 2 and $S \subseteq E(\overline{\mathbb{F}}_{p^2})$ consiting points of odd order. $\mathsf{GenIdealToIsogenyIQO}$ then returns $E, I'', \varphi_{I''}(\tilde{P}_0, \tilde{Q}_0), \varphi_{I'}(S_{odd})$ where E is the codomain of $\varphi_{I'}$, I'' is another ideal equivalent to I' with odd norm, $\tilde{P}_0, \tilde{Q}_0$ is some fixed basis of $E_0[2^e]$.

Remark 44. The only additional output of $\mathsf{GenIdealToIsogenyIQO}$ is $\varphi_{I'}(S_{odd})$. This modification is not hard to achieve. In $\mathsf{IdealToIsogenyIQO}$, the algorithm essentially decomposes $\varphi_{I'}$ into an isogeny chain and each isogeny in the composition is computed via Vélu's formulas. Therefore this evaluation $\varphi_{I'}(S_{odd})$ can also be computed at the same time when each isogeny is computed.

Proposition 45. *Algorithm 1 is correct.*

Proof. By design, γ satisfies that $\gamma \in I$ and $I'' = I\bar{\gamma}/n(I)$. According to [5, Lemma 9], $\hat{\varphi}_{I'} \circ \varphi_I = \iota_0(\alpha), \hat{\varphi}_{I''} \circ \varphi_I = \iota_0(\gamma)$. Therefore,

Algorithm 1: IdealToIsogeny

Input: I, S, N where

1. I is a left $\mathcal{O}_0$-ideal of odd prime norm,
2. $S = S_2 + S_{odd} \subseteq E(\bar{\mathbb{F}}_{p^2})$ where $S_2 \subseteq E_0[2^e]$ and S_{odd} consists of points whose orders are odd,
3. an optional input N such that N is a prime factor of $n(I)$.

Output: codomain E of φ_I, $\varphi_I(S)$ and a generator of $\hat{\varphi}_I(E[N])$ when the optional input N is given

1 Use the KLPT algorithm [29] to compute $\alpha \in I$ such that $n(\alpha) = 2^n n(I)$.
2 $I' \leftarrow I\bar{\alpha}/n(I)$.
3 $E, I'', P_{I''}, Q_{I''}, \varphi_{I'}(\iota_0(\alpha)(S_{odd})) \leftarrow$ GenIdealToIsogenyIQO$(\mathcal{O}_0, I', \iota_0(\alpha)(S_{odd}))$.
4 Compute $\bar{\beta} \in \mathcal{O}_0$ that is a generator of the principal ideal left $\mathcal{O}_R(I')$-ideal $\bar{I'}I''$.
5 $\gamma \leftarrow \beta\alpha/n(I')$.
6 $P_{0,\gamma}, Q_{0,\gamma} \leftarrow \iota_0(\gamma)(\tilde{P}_0, \tilde{Q}_0)$.
7 Compute $m_{11}, m_{12} \in \mathbb{Z}/2^e\mathbb{Z}$ such that $P_{0,\gamma} = [m_{11}]\tilde{P}_0 + [m_{12}]\tilde{Q}_0$.
8 Compute $m_{21}, m_{22} \in \mathbb{Z}/2^e\mathbb{Z}$ such that $Q_{0,\gamma} = [m_{21}]\tilde{P}_0 + [m_{22}]\tilde{Q}_0$.
9 $m \leftarrow (n(I'') \bmod 2^e)^{-1}$.
10 $P \leftarrow [m]([m_{11}]P_{I''} + [m_{12}]Q_{I''})$.
11 $Q \leftarrow [m]([m_{21}]P_{I''} + [m_{22}]Q_{I''})$.
12 $S' = \{\}$.
13 **for** $P_2 \in S_2$ **do**
14 Compute x_1, x_2 such that $P_2 = [x_1]\tilde{P}_0 + [x_2]\tilde{Q}_0$.
15 $S' = S' + \{[x_1]P + [x_2]Q\}$
16 **end**
17 $S' = S' + [n(I')^{-1}]\varphi_{I'}(\iota_0(\alpha)(S_{odd}))$. \\ The inverse of $n(I')$ is taken modulo the order of points in S_{odd} which already exists since $n(I')$ is a power of 2
18 **if** present(N) **then**
19 Compute a generator K of $\iota_0\hat{(}\alpha)(E_0[N])$.
20 **return** E, S', K
21 **else**
22 **return** E, S'
23 **end**

- $[n(I')]\hat{\varphi}_I = \iota_0\hat{(}\alpha) \circ \hat{\varphi}_{I'}$, from this it follows that $\hat{\varphi}_I(E[N]) = \iota_0\hat{(}\alpha)(E_0[N])$ since $n(I')$ is a power of 2 and is coprime to N.
- Equivalently, we have $[n(I')]\varphi_I = \varphi_{I'} \circ \iota_0(\alpha)$, from this the correctness of the evaluations $\varphi_I(S_{odd})$ follows.
- $[n(I'')]\varphi_I = \varphi_{I''} \circ \iota_0(\gamma)$, from this the correctness of the evaluations $\varphi_I(S_2)$ follows since $n(I'')$ is odd.

□

Remark 46. For our applications, the most time-consuming step in Algorithm 1 will be Line 3 where 2-dimension isogeny computations are needed. Specifically,

let $e = e_1 + e_2$ and $2^{e_2} \approx \sqrt{p}$, then IdealToIsogenyIQO needs to operate $3 \log p / e_1$ many $(2^{e_2}, 2^{e_2})$-isogeny computations. The numerator $3 \log p$ comes from the fact that $n(\alpha)/n(I) \approx p^3$ where α is the output of Line 2.

Remark 47. There are other ideal to isogeny algorithms such as the ones from [5, 14] which are more efficient than [33]. However, the algorithms from [5, 14] requires that the set of rational points $E(\mathbb{F}_{p^2})$ contains as much 2-power torsions as possible, and this leaves no room for having the delay torsion T also defined over $\mathbb{F}_{p^2}$. Moving T to $E^t(\mathbb{F}_{p^2})$ (E^t denotes the quadratic twist of E) is a potential solution but it poses significant challenges in generating parameters for any given delay parameter t as each time one needs to factor $p - 1$. Therefore, we choose to adapt the ideal to isogeny algorithm from [33].