Verifiable Delay Functions from Supersingular Isogenies and Pairings

Luca De Feo[1][0000−0002−9321−0773], Simon Masson[2], Christophe Petit[3], and Antonio Sanso[4]

[1] Université Paris Saclay – UVSQ, LMV, CNRS UMR 8100, Versailles, FR.
https://defeo.lu/
[2] Thales and Université de Lorraine
[3] University of Birmingham
[4] Adobe Inc. and Ruhr Universität Bochum

Abstract. We present two new Verifiable Delay Functions (VDF) based on assumptions from elliptic curve cryptography. We discuss both the advantages and drawbacks of our constructions, we study their security and we demonstrate their practicality with a proof-of-concept implementation.

1 Introduction

A Verifiable Delay Function (VDF), first formalized in 2018 by Boneh, Bonneau, Bünz and Fisch [7], is a function $f : X \to Y$ that takes a prescribed *wall-clock time* to evaluate, independently of the parallelism of the architecture employed, and such that its output can be verified efficiently. In a nutshell, it is required that anyone can evaluate f in T *sequential steps*, but no less, even on a large number of processors; on top of that, given an input x and an output y, anyone must be able to verify that $y = f(x)$ in a *short* amount of time, desirably in $\mathrm{polylog}(T)$.

An example of a VDF lacking efficient verification is a chained one-way function:

$$s \to H(s) \to H(H(s)) \to \cdots \to H^{(T)}(s) = a.$$

This clearly takes T *steps* to evaluate, even on a parallel computer, however the only feasible way to verify the output is to re-evaluate the function. Two related known crypto primitives are the time-lock puzzles defined by Rivest, Shamir, and Wagner in [55] and proofs of sequential work [44,18]. The problem with the former is that it is not publicly verifiable while the latter is not a function (i.e., it does not have a unique output).

A VDF based on univariate permutation polynomials over finite fields is presented in [7], along with other candidate constructions, none being entirely satisfactory (see next section). The same work listed as an open problem to find theoretically optimal VDFs based on simple assumptions closer to those typically found in other asymmetric protocols. Pietrzak [50] and Wesolowski [64] responded to the challenge by proposing two practical VDFs based on exponentiation in a group of unknown order. Both VDFs are surveyed in [8].

Our contribution. We present a new framework for VDFs, and two instantiations of this framework using isogenies of supersingular elliptic curves and bilinear pairings. Both our constructions are optimal and unconditionally sound. We observe that the construction based on univariate permutation polynomials of Boneh et al. [7] also has both properties, but its security relies on an *ad hoc* limit assumption on the amount of parallelism available to the attacker. Moreover unlike the VDF constructions of Pietrzak [50] and Wesolowski [64], ours are inherently non-interactive, the output being efficiently verifiable without attaching a proof. By using mathematical tools also used in other cryptographic contexts, our constructions benefit from pre-existing research in these areas both from an efficiency and security point of view. Finally, while the use of isogenies does not magically make our functions post-quantum (in fact they can be broken with a discrete logarithm computation), our second proposal still offers some partial resistance to quantum attacks; we call this property *quantum annoyance*.

The main drawback of our proposals is that, given current knowledge, the only secure way to instantiate our VDFs requires a *trusted setup*, or, said otherwise, that our VDFs can be easily backdoored. Indeed, both our setups require to start from a supersingular elliptic curve with unknown endomorphism ring. No general algorithm is known to compute the endomorphism ring of supersingular elliptic curves, however the only known ways to generate supersingular curves involve a random isogeny walk from a curve with small discriminant (e.g., $j = 0$ or $j = 1728$), and it has been shown that knowledge of the isogeny walk permits computing the endomorphism ring in polynomial time [39,27]. Hence, unless a way is found to generate random supersingular curves with unknown endomorphism ring, the only way to instantiate our VDFs involves a trusted setup that performs a random isogeny walk and then forgets it.

Another minor drawback of our VDFs is that the time required to setup public parameters is of the same order of magnitude as that required to evaluate the function; furthermore, validating public parameters requires the same amount of time as evaluating the function, and the evaluator is required to use $O(T)$ storage for evaluating in optimal time. We propose some partial mitigations to these problems in our implementation, however further research is needed to address them completely.

Finally, our proposals can be seen as a generalization of BLS signatures [9], where the secret scalar is replaced by a walk in an isogeny graph. By keeping the walk secret, we obtain a signature/identification scheme with very similar properties to BLS, with the added benefit of offering a partial form of resistance to quantum attacks: while forgeries are possible using Shor's algorithm, key recovery seems to be infeasible even with quantum computers.

This paper is organized as follows. In Section 2 we formalize Verifiable Delay Functions and we go through some of the proposed solutions. Section 3 to Section 5 provide a description of our VDFs, together with a review of the basic theory of supersingular isogeny graphs. Section 6 gives security proofs and reviews the available attacks against our proposals. Finally 7 provides an optimized implementation of our VDFs, and related benchmarks.

2 Verifiable Delay Functions

We recall here the formal definition of a Verfiable Delay Function, following [7]. A VDF consists of three algorithms:

1. $\mathsf{Setup}(\lambda, T) \rightarrow (\mathsf{ek}, \mathsf{vk})$: is a procedure that takes a security parameter λ, a delay parameter T, and outputs public parameters consisting of an evaluation key ek and a verification key vk.

2. $\mathsf{Eval}(\mathsf{ek}, s) \rightarrow (a, \pi)$: is a procedure to evaluate the function on input s. It produces the output a from s, and a (possibly empty) proof π. This procedure is meant to be infeasible in time less than T.

3. $\mathsf{Verify}(\mathsf{vk}, s, a, \pi) \rightarrow \{\mathsf{true}, \mathsf{false}\}$: is a procedure to verify that a is indeed the correct output for s, with the help of the proof π.

A VDF shall satisfy three security properties: *Correcteness*, stating that a honest evaluator always passes verification, *Soundness*, stating that a lying evaluator never passes verification, and *Sequentiality*, stating that it is impossible to correctly evaluate the VDF in time less than $T - o(T)$, even when using $\mathrm{poly}(T)$ parallel processors. We will give formal security definitions in Section 6, but see also [7].

According to [7], the Setup routine should run in time $\mathrm{poly}(\lambda)$; here we slightly relax this constraint and allow it to run in $\mathrm{poly}(T, \lambda)$. Eval must be doable in time $T - o(T)$; Verify in time $\mathrm{poly}(\lambda)$. A VDF is said to be *optimal* when T is allowed to be in $o(2^\lambda)$ without harming security; note that it does not make sense to have $T \in O(2^\lambda)$, since in that case it is cheaper to break soundness than to run Eval.

VDF have several applications. Among them, we cite:

– Constructing a trustworthy ***randomness beacon***, like the one introduced by Rabin in [53], where a public service produces a continuous stream of guaranteed unbiased randomness. The classic approach consisting in extracting randomness from entropy pool sources, such as stock prices or proof-of-work blockchains *à la* Bitcoin, has been shown to be manipulable by active attackers [49]. For example, while the price of a particular stock may seem unpredictable to a passive observer, a powerful trader can influence the market trend, making the random output biased. Here is where VDFs are useful: if the beacon is calculated by applying a VDF with a long enough delay to the entropy source, the malicious trader would not have the time to try to "adjust" the market at his own advantage.
The other common solution based on the "commit-and-reveal" paradigm with ***multiparty randomness*** has also been shown to have flaws. Indeed a malicious party with the intention of manipulating the output might refuse to reveal his commitment after seeing the other opened commitments, forcing to restart the protocol. In this case too, as shown by Lenstra and Wesolowski [43], the usage of VDFs instead of commitments helps fix the multiparty randomness beacon service.

– VDFs may be used to reduce the energy consumption of blockchains based on proofs-of-work. An elegant idea by Cohen [16] combines proofs-of-resources with incremental VDFs in order to achieve ***Consensus from Proof of Resources***. In particular, he describes a technique based on proof of space where the mining reward is roughly equal to the value of the space owned, without each miner running a large parallel computation. At high level this works as follows. Suppose a miner controls N units of the total space and splits his proof π (proving control of the N units) into N pieces $\pi_1, \pi_2 ... \pi_N$. The miner then computes $H_i = \text{HASH}(\pi_i) \in [0, N]$ and $\tau = \min(H_1, ..., H_N)$. At this point the miner evaluates a VDF with time delay proportional to τ. The first miner that successfully computes the output is the "winner" and has the block assigned. For a miner that controls N units of the total space this will happen with a probability that is about $N\%$. The Chia blockchain [47] has been designed to work with this model.

For a description of other applications of VDFs, such as **proof of replication** or **computational timestamping**, we refer to [7]. So far, few constructions meet the requirements of a VDF; we summarize them below.

Modular square roots. One of the earlier examples of VDF can be found in the 1992 paper by Dwork and Naor [26]. The underlying idea is rather simple: given a prime number p such that $p = 3 \bmod 4$, a (canonical) square root $a = \sqrt{s} \bmod p$ can be computed using the formula $a = s^{\frac{p+1}{4}}$. This requires about $\log(p)$ sequential squaring operations. On the other hand, verifying correctness only requires to check that $a^2 = s$. While there is a gap between the evaluation and the verification operations, this simple approach has two issues: first, the gap is only polynomial in the delay parameter $T = \log(p)$, secondly, due to the possibility of parallelizing field multiplications, this gap vanishes asymptotically if the evaluator is provided large amounts of parallelism (see also Table 1). Lenstra and Wesolowski introduced with Sloth [43] the possibility of chaining square root operations. The problem with this construction, though, is that it does not achieve asymptotically efficient verification.

Time-lock puzzles. Time-lock puzzles were introduced by Rivest, Shamir, and Wagner [55] to provide encryption that can only be decrypted at a set time in the future. They use a classical RSA modulus $N = pq$; the encryption key is then $a = s^{2^T} \bmod N$ for some starting value s. Now, it is clear that any party knowing $\varphi(N)$ can compute the value of a quickly (they can reduce the exponent $e = 2^T \bmod \varphi(N)$). But for everyone else the value of a is obtained by computing T sequential squaring operations.

The main reason why this construction cannot be classified as a VDF is that there is not an efficient way to perform public verification without giving away the factorization of N. This issue has recently been solved, independently, by Pietrzak and Wesolowski. We briefly present their constructions next; for a more in-depth survey, see [8].

Wesolowski's VDF. In 2018, Wesolowski presented a VDF based on groups of unknown order [64].[5] His work leverages the time lock puzzle described above, introducing a way to publicly verify the output $a = s^{2^T}$. He defines an interactive protocol where, after seeing the output a, the verifier sends to the prover a random prime $\ell < B$, where B is some small bound. The prover replies with the value $b = s^{\lfloor 2^T/\ell \rfloor}$; the verifier then checks that $a = b^\ell s^r$, where $r = 2^T \bmod \ell$. Because the verifier only uses public randomness, this protocol can be made non-interactive using the Fiat–Shamir heuristic. Wesolowski's proposal shines for the shortness of the proof (only one group element) and the speed of the verification (only two group exponentiations).

Wesolowski suggests two ways of instantiating groups of unknown order. The first one is using RSA groups $(\mathbb{Z}/N\mathbb{Z})^*$, like in Rivest–Shamir–Wagner, and thus requires a trusted third party to produce the modulus N. The second is using class groups of imaginary quadratic number fields [12,17]. While the former instantiation is better studied in public key cryptography, the second has the advantage of not requiring a trusted setup.

Pietrzak's VDF. Concurrently with Wesolowski, Pietrzak [50] introduced another protocol to verify Rivest–Shamir–Wagner time-lock puzzles. Pietrzak's verification procedure is an interactive recursive protocol, where the prover outputs a proof π consisting of $O(\log(T))$ group elements, and the verifier needs about $O(\log(T))$ time to do the verification. The main advantage of his construction is that the prover only needs about $O(\sqrt{T})$ group multiplications to build π. Pietrzak presents his protocol using RSA groups, but class groups like in Wesolowski's VDF can also be used (although this affects slightly the computational assumptions needed for soundness).

Univariate permutation polynomials. Boneh, Bonneau, Bünz and Fisch explored in their seminal paper [7] an approach based on permutation polynomials over finite fields $\mathbb{F}_p$. In full generality, their proposal is a weaker form of VDF, where a certain amount of parallelism is needed to give an advantage to the evaluator (see [7, Definition 5]). The gist of their approach is that, given a permutation polynomial of degree T, inverting such polynomial implies computing polynomial GCDs. This operation takes $O(\log(p))$ multiplications of dense polynomials of degree $O(T)$, and it is conjectured that it cannot be done in less than T steps on at most $O(T^2)$ processors (see [7, Assumption 2]). On the other hand, any such polynomial can be evaluated, and thus verified, using $O(\log(T))$ operations on $O(T)$ processors, which is exponentially smaller. Moreover, there exists a family of permutation polynomials, due to Guralnick and Müller [32], that can be evaluated in $O(\log(T))$ operations without parallelism, and it is conjectured in [7] that the derived VDF is secure.

The drawbacks of this construction are that the parallelism of the evaluator needs to be polynomially bounded in T, and that it is based on assumptions that have been seldom studied in a cryptographic setting.

[5] Wesolowski also introduces *trapdoor verifiable delay functions.*

VDF	Sequential Eval	Parallel Eval	Verify	Setup	Proof size
Modular square root	T	$T^{2/3}$	$T^{2/3}$	T	—
Univariate permutation polynomials[6]	T^2	$> T - o(T)$	$\log(T)$	$\log(T)$	—
Wesolowski's VDF	$(1 + \frac{2}{\log(T)})T$	$(1 + \frac{2}{s\log(T)})T$	λ^4	λ^3	λ^3
Pietrzak's VDF	$(1 + \frac{2}{\sqrt{T}})T$	$(1 + \frac{2}{s\sqrt{T}})T$	$\log(T)$	λ^3	$\log(T)$
This work	T	T	λ^4	$T\lambda^3$	—
This work (optimized)	T	T	λ^4	$T\log(\lambda)$	—

Table 1. VDF comparison—Asymptotic VDF comparison: T represents the delay factor, λ the security parameter, s the number of processors. For simplicity, we assume that T is super-polynomial in λ. All times are to be understood up to a (global across a line) constant factor.

Incrementally Verifiable SNARK For completeness we need to mention that a theoretical, albeit impractical, VDF can be constructed using Incrementally Verifiable SNARKs. Again, we refer to [7] for a deeper analysis of the topic.

We compare the asymptotic performance of the VDFs above and of our proposal in Table 1. Outside of modular square roots, all VDFs constructions meet the requirements of an optimal VDF, however each has its qualitative strengths and weaknesses: permutation polynomials require to bound the parallelism of the evaluator, and are based on little studied assumptions; VDFs derived from time-lock puzzles are interactive, have no unconditional soundness, and may or may not require a trusted setup; ours need a trusted setup, and require an effort to validate public parameters comparable to evaluating the VDF.

3 Abstract description

We start by describing a framework for defining VDFs inspired by the BLS signature scheme based on pairing groups [9]. Recall that BLS uses a *pairing friendly* elliptic curve $E/\mathbb{F}_p$, with a non-degenerate bilinear pairing $e_N : X_1 \times X_2 \to \mathbb{F}_{p^k}$, where X_1, X_2 are subgroups of prime order N, and the extension degree k is called the *embedding degree*. The secret key in BLS is a scalar $s < N$, and the public key is a pair of points $P, sP \in X_1$. To sign a message m, the signer computes a hash $Q = H(m) \in X_2$, and gives back the signature sQ. The verifier then checks that $e_n(P, sQ) = e_n(sP, Q)$.

The BLS signature is also naturally a Verifiable Random Function $f_s : X_2 \to X_2$, where only the owner of the trapdoor s can evaluate f_s, while anyone can

[6] According to [7, § 5.1], one must limit the evaluator to $O(T^2)$ parallel processors for the bound on parallel Eval to hold. VDFs based on permutation polynomials can be evaluated in time $O\big(\log^2(T)\big)$ using $O(T^{3.8})$ parallel processors.

verify the result; however, it is not a VDF, because both evaluation and verification are in polylog(N). Our generalization, instead, has efficient instantiations based on isogeny graphs of supersingular elliptic curves, where the evaluation can be made exponentially slower than the verification. If the trapdoor is kept secret, one obtains a signature/identification protocol based on walks in isogeny graphs; if the trapdoor is made public, one obtains a VDF.[7] Our setup is also related to the Isogenous Pairing Groups framework of Koshiba and Takashima [40], who generalize the Boneh-Franklin IBE with the goal of adding some (weak) quantum resistance. We will present our instantiations in Section 5.

Let X_1, X_2, Y_1, Y_2, G be groups of prime order N, let $e_X : X_1 \times X_2 \to G$ and $e_Y : Y_1 \times Y_2 \to G$ be non degenerate bilinear pairings. Furthermore, assume that there is a pair of bijections $\phi : X_1 \to Y_1$ and $\hat{\phi} : Y_2 \to X_2$ that satisfy the following diagram,

$$\begin{array}{ccc} X_1 \times Y_2 & \xrightarrow{\phi \times 1} & Y_1 \times Y_2 \\ {\scriptstyle 1 \times \hat{\phi}} \downarrow & & \downarrow {\scriptstyle e_Y} \\ X_1 \times X_2 & \xrightarrow[e_X]{} & G \end{array}$$

Note that the diagram implies ϕ and $\hat{\phi}$ are group isomorphisms.

We shall assume that the pairings e_X, e_Y can be evaluated in time polylog(N), whereas both ϕ and $\hat{\phi}$ can be evaluated in sequential time T, where T is some parameter independent from N (but still in $o(N)$).

Let P be any generator of X_1, the public parameters of our system are going to be $(N, X_1, X_2, Y_1, Y_2, G, e_X, e_Y, P, \phi(P))$. From this setup, we derive two primitives:

An identification protocol. The maps ϕ and $\hat{\phi}$ are the trapdoor. The verifier gives an element $Q \in Y_2$ to the prover, the proof is the element $\hat{\phi}(Q)$. Then, the verifier checks that

$$e_X(P, \hat{\phi}(Q)) = e_Y(\phi(P), Q).$$

It should be apparent that BLS signatures correspond to the special case where $X_1 = Y_1$ and $X_2 = Y_2$ are orthogonal groups with respect to an elliptic pairing $e_X = e_Y$, and $\phi = \hat{\phi} = [s]$ is the multiplication endomorphism by a secret scalar s.

The same abstract scheme already appears in a patent by Broker, Charles and Lauter [11], although their implementation is different, and likely less efficient. We shall see in Section 6 that our instantiation presents the minor advantage over BLS signatures of being partially resistant to quantum attacks.

[7] Note that this is different from a *trapdoor VDF*, as defined by Wesolowski [64], where the trapdoor is used to efficiently compute the evaluation.

A VDF. The maps ϕ and $\hat{\phi}$ are also part of the public parameters. The VDF is the map $\hat{\phi}$, Eval simply amounts to evaluating it at points $Q \in Y_2$. To verify the output, one checks that

$$e_X(P, \hat{\phi}(Q)) = e_Y(\phi(P), Q).$$

It should be clear that, because the map $R \mapsto e_X(P, R)$ is an isomorphism, verification succeeds if and only if the output is correct; this will be used to prove *correctness* and *soundness* in Section 6. By hypothesis, Eval takes T sequential steps, while the pairings can be evaluated in time $\mathrm{polylog}(N)$.

4 Preliminaries on supersingular curves

Before describing the instantiations, we review some basic facts on supersingular curves, pairings and isogenies. For details on elliptic curves over finite fields see [56,62,57], for their use in cryptography see [6,29,21], for ideal class groups of quadratic imaginary fields see [20], for maximal orders of quaternion algebras see [60,61].

Let E be an elliptic curve defined over a finite field $\mathbb{F}_q$ of characteristic p. Recall that the order of $E(\mathbb{F}_q)$ is $\#E(\mathbb{F}_q) = q + 1 - t$, where t is the *trace* of the *Frobenius endomorphism* π. Then, a curve is *supersingular* if and only if p divides t. Every supersingular curve is isomorphic to a curve defined over $\mathbb{F}_{p^2}$, so, for a fixed prime p, there is only a finite number of supersingular curves, up to isomorphism.

An *isogeny* of E is an algebraic group morphism from E to some other curve E'. For *separable* isogenies,[8] the *degree* is the size of their kernel; isogenies of degree ℓ are called ℓ-isogenies. A separable isogeny is said to be *cyclic* if its kernel is; we will mostly deal with cyclic isogenies in this work.

For any ℓ-isogeny $\phi : E \to E'$, there is a unique ℓ-isogeny $\hat{\phi} : E' \to E$, called the *dual* of ϕ, such that $\phi \circ \hat{\phi} = [\ell]$ on E' and $\hat{\phi} \circ \phi = [\ell]$ on E. This shows that being *ℓ-isogenous* is a symmetric relation, and that being isogenous is an equivalence relation. Further, a theorem of Tate states that two curves are isogenous over $\mathbb{F}_q$ if and only if they have the same number of points over $\mathbb{F}_q$, thus in particular a supersingular curve can only be isogenous to other supersingular curves.

One can define several bilinear pairings on supersingular curves. In this paper we will use the Weil pairing $e_N : E[N] \times E[N] \to \mu_N$ for describing the protocol, although the (reduced) Tate pairing is better suited for implementation purposes. The pairings will have embedding degree 2 or 1, depending on the VDF, construction. Most importantly, both pairings will satisfy the compatibility condition

$$e_N(\phi(P), Q) = e_N(P, \hat{\phi}(Q))$$

[8] An isogeny is separable if it induces a separable extension of function fields. We will only use separable isogenies in this work.

for any isogeny $\phi : E \to E'$ and points $P \in E[N]$, $Q \in E'[N]$. See [6, Chapters IX–X] for more details.

Graphs of supersingular isogenies have been studied by Mestre [46], Pizer [51,52], Kohel [38], Delfs and Galbraith [24], among others. We distinguish two important families: graphs of ℓ-isogenies and curves *defined over a prime field* $\mathbb{F}_p$ (i.e., expressed by rational fractions with coefficients in $\mathbb{F}_p$), and graphs of ℓ-isogenies defined over the algebraic closure $\overline{\mathbb{F}}_p$ (or, equivalently, over $\mathbb{F}_{p^2}$). In the following, we shall assume that $p > 3$.

Graphs over $\mathbb{F}_p$. For the first case, Delfs and Galbraith showed that one obtains the same kinds of undirected graphs as for ordinary curves. In this case $t = 0$ and $\#E(\mathbb{F}_p) = p + 1$, thus there is a unique *isogeny class* containing all supersingular curves defined over $\mathbb{F}_p$. To give a more precise classification, following Kohel [38], we say that an isogeny $\phi : E \to E'$ is *horizontal* whenever $\mathrm{End}_{\mathbb{F}_p}(E) \simeq \mathrm{End}_{\mathbb{F}_p}(E')$; Delfs and Galbraith showed that there are one or two horizontal isogeny classes of supersingular curves over $\mathbb{F}_p$, according to whether $p = \pm 1 \bmod 4$. Precisely:

- If $p = 1 \bmod 4$, then $\mathrm{End}(E) \simeq \mathbb{Z}[\sqrt{-p}]$ for all curves, and the isogeny class contains h curves, up to $\mathbb{F}_p$-isomorphism, where h is the *class number* of the imaginary quadratic field $\mathbb{Q}(\sqrt{-p})$.
- If $p = -1 \bmod 4$, then $\mathrm{End}(E)$ is isomorphic to one of $\mathbb{Z}[\sqrt{-p}]$ or $\mathbb{Z}[(1 + \sqrt{-p})/2]$; the horizontal isogeny class associated to $\mathbb{Z}[(1 + \sqrt{-p})/2]$ is called the *surface* and contains h curves up to $\mathbb{F}_p$-isomorphism; the horizontal isogeny class associated to $\mathbb{Z}[\sqrt{-p}]$ is called the *floor*, and contains h or $3h$ curves, according to whether $p = 7 \bmod 8$ or $p = 3 \bmod 8$ respectively.

The connectivity of the ℓ-isogeny graphs will now depend on the chosen degree ℓ. Specifically:

- If ℓ is an odd prime and $\left(\frac{-p}{\ell}\right) = -1$ (i.e., $-p$ is not a square modulo ℓ), no ℓ-isogeny of supersingular curves is defined over $\mathbb{F}_p$, i.e., the ℓ-isogeny graph is made of isolated vertices.
- If ℓ is an odd prime and $\left(\frac{-p}{\ell}\right) = 1$, every curve has exactly two horizontal ℓ-isogenies, thus each horizontal isogeny class is partitioned into a finite number of cycles.
- If $\ell = 2$ and $p = 1 \bmod 4$, then every curve has exactly one horizontal ℓ-isogeny.
- If $\ell = 2$ and $p = -1 \bmod 4$, then every curve on the floor has exactly one non-horizontal ℓ-isogeny going to a curve on the surface, whereas for curves on the surface:
  - If $p = 7 \bmod 8$, they have exactly two horizontal ℓ-isogenies, plus one non-horizontal going to the floor, dual to the one coming from the floor;
  - If $p = 3 \bmod 8$, they have three non-horizontal isogenies going to three curves on the floor, dual to the ones coming from the floor.

In the rest of this work, we will only be interested in cycles of horizontal isogenies, thus either ℓ odd and $\left(\frac{-p}{\ell}\right) = 1$, or $\ell = 2$, $p = 7 \bmod 8$ and the curves on the surface.

Graphs over $\mathbb{F}_{p^2}$. Over $\mathbb{F}_{p^2}$ there is more than one isogeny class, indeed the trace t of a supersingular curve can take any of the values $0, \pm p, \pm 2p$. The values $t = 0$ and $t = \pm p$ produce exceptional classes made of only one element, and are thus not interesting for cryptography. The cases $t = \pm 2p$ produce two distinct classes, each with $\lfloor p/12 \rfloor + c_p$ elements, where $0 \leq c_p \leq 2$ is a constant depending only on p mod 12; these two classes are isomorphic in the sense that each curve in one is $\bar{\mathbb{F}}_p$-isomorphic to exactly one curve in the other, and thus we typically speak of supersingular graphs over $\bar{\mathbb{F}}_p$ and over $\mathbb{F}_{p^2}$ indistinctly.

For any prime $\ell \neq p$, the ℓ-isogeny graph of supersingular curves over $\mathbb{F}_{p^2}$ is an $(\ell+1)$-regular multi-graph, undirected outside of the two special vertices $j = 0, 1728$. In $\mathbb{F}_{p^2}$ we do not encounter the concept of horizontal isogenies anymore: every endomorphism ring is isomorphic to a maximal order in a quaternion algebra, and to every maximal order corresponds exactly a pair of ($\mathbb{F}_{p^2}/\mathbb{F}_p$-Galois conjugate) supersingular curves.

It is still true, however, that we may find inside the graph a sub-structure inherited from the graph of supersingular curves defined over $\mathbb{F}_p$. One may be tempted to think that the $\mathbb{F}_{p^2}$-graph contains the $\mathbb{F}_p$-graph as a subgraph, however the situation is slightly subtler: indeed, supersingular curves over $\mathbb{F}_p$ are isogenous to their quadratic twists, thus the $\mathbb{F}_p$-graph contains pairs of vertices that become isomorphic in $\mathbb{F}_{p^2}$. Hence, the ℓ-isogeny graph of curves and isogenies defined over $\mathbb{F}_p$ is a double cover (outside the ramification points at $j = 0, 1728$) of the $\mathbb{F}_p$-subgraph contained in the ℓ-isogeny graph over $\mathbb{F}_{p^2}$. Fear not: this technical detail will be completely irrelevant to us.

5 Two instantiations with supersingular elliptic curves

We now give two instantiations of the VDF described in Section 3, using supersingular elliptic curves for the pairing groups, and isogenies of prime power degree for the maps $\phi, \hat{\phi}$. We will see in Section 6 that the choice of the curves severely affects the security of the protocol, however we ignore this issue for the moment. In this section we will describe the VDFs using the Weil pairing, however for implementation purposes we will use the Tate pairing in Section 7.

5.1 VDF from supersingular curves over $\mathbb{F}_p$

Our first construction uses supersingular curves defined over a prime field $\mathbb{F}_p$. It shares similarities with the key exchange protocol CSIDH [13] and with the VDF based on class groups of imaginary quadratic fields by Wesolowski [64].

Let p be a prime such that $p + 1$ contains a large prime factor N. Let ℓ be one of:

- $\ell = 2$, only if $p = 7 \bmod 8$, or
- a small prime such that $\left(\frac{-p}{\ell}\right) = 1$.

Let $E/\mathbb{F}_p$ be a supersingular elliptic curve, and denote by $e_N(\cdot, \cdot)$ the Weil pairing on $E[N]$. When $\ell = 2$ we shall add the requirement that $E[2] \subset E(\mathbb{F}_p)$,

implying that E is on the surface. By construction $\#E(\mathbb{F}_p) = p+1$, and $E(\mathbb{F}_p)$ contains exactly one cyclic subgroup of order N, that we shall use as $X_2 = E[N] \cap E(\mathbb{F}_p)$.

Let $u \in \mathbb{F}_p$ be any non quadratic residue. We define a map

$$\begin{aligned} \upsilon : E &\to \tilde{E} \\ (x, y) &\mapsto (u^2 x, u^3 y) \end{aligned}$$

to a *quadratic twist* $\tilde{E}$ of E, i.e., to a curve that is isomorphic to E over $\mathbb{F}_{p^2}$ but not over $\mathbb{F}_p$. By construction, $\tilde{E}$ has the same order $\#\tilde{E}(\mathbb{F}_p) = p+1$, and it contains exactly one cyclic subgroup $\tilde{X}_1 = \tilde{E}[N] \cap \tilde{E}(\mathbb{F}_p)$; we shall then set X_1 to $\upsilon^{-1}(\tilde{X}_1)$. Finally, the restriction of the Weil pairing to $X_1 \times X_2$ is non-degenerate, as wanted.[9]

The map ϕ will be instantiated with an isogeny of degree ℓ^T, and the map $\hat{\phi}$ with its dual. In practice, we assume that these isogenies are stored as a sequence of T isogenies of degree ℓ (e.g., specified by their kernels), so that evaluating ϕ and $\hat{\phi}$ can be done in time polynomial in ℓ and linear in T. For a representation that is more compact by a (large) constant factor, see Section 7.

Because of the way we have chosen ℓ, the graph of (horizontal) ℓ-isogenies containing E is a cycle of length dividing the class number h, thus an isogeny of degree ℓ^T is obtained by choosing a *direction* on the cycle and composing T isogeny steps each of degree ℓ. The isogeny $\phi : E \to E'$ defines an image curve $E'/\mathbb{F}_p$ having the same group structure as E; in particular we define the cyclic groups $Y_1 = \upsilon^{-1}(\tilde{E}'[N] \cap \tilde{E}'(\mathbb{F}_p))$ and $Y_2 = E'[N] \cap E'(\mathbb{F}_p)$, where $\tilde{E}' = \upsilon(E')$ is a quadratic twist of E'.

Note that it is easy to sample uniformly from any of the groups X_1, X_2, Y_1, Y_2, in a way that does not reveal discrete logarithms:[10] one simply takes random points on the curves or on their twists and multiplies by the cofactor $(p+1)/N$. The algorithms defining the VDF are described in Figure 1.

The similarity with Wesolowski's VDF is evident here: all ℓ-isogenies with the same direction correspond to an ideal $\mathfrak{a}$ of norm ℓ inside the quadratic imaginary order $\mathcal{O} \simeq \mathrm{End}_{\mathbb{F}_p}(E)$, which is also a representative of an ideal class in $\mathrm{Cl}(\mathcal{O})$. Composing isogenies corresponds to multiplying ideals, thus ϕ corresponds to $\mathfrak{a}^T$ and $\hat{\phi}$ corresponds to $\mathfrak{a}^{-T}$ in $\mathrm{Cl}(\mathcal{O})$. While Wesolowski raises elements to the power 2^T, we only do the equivalent of raising to the power T, because no analogue of the square-and-multiply algorithm is known for composing isogenies. Of course, the fundamental difference is in the way we verify the computation.

[9] We note that a distorsion map $X_1 \to X_2$ may be used to define a self-pairing on X_1, however efficient distortion maps only exist for very few supersingular curves. Fortunately, we will not need distorsion maps.

[10] In the elliptic curve cryptography literature, this is typically called *hashing* into the groups.

$\mathsf{Setup}(\lambda, T)$

1. Choose primes N, p with the properties above, according to the security parameter λ;
2. Select a supersingular curve $E/\mathbb{F}_p$;[11]
3. Choose a direction on the horizontal ℓ-isogeny graph, and compute a cyclic isogeny $\phi : E \to E'$ of degree ℓ^T, and its dual $\hat{\phi}$;
4. Choose a generator P of $X_1 = \upsilon^{-1}(\tilde{E}[N] \cap \tilde{E}(\mathbb{F}_p))$, and compute $\phi(P)$;
5. Output $(\mathsf{ek}, \mathsf{vk}) = \big(\hat{\phi}, (E, E', P, \phi(P))\big)$.

$\mathsf{Eval}(\hat{\phi}, Q \in Y_2)$

1. Compute and output $\hat{\phi}(Q)$.

$\mathsf{Verify}(E, E', P, Q, \phi(P), \hat{\phi}(Q))$

1. Verify that $\hat{\phi}(Q) \in X_2 = E[N] \cap E(\mathbb{F}_p)$;
2. Verify that $e_N(P, \hat{\phi}(Q)) = e_N(\phi(P), Q)$.

Fig. 1. Instantiation of the Verifiable Delay Function over $\mathbb{F}_p$

5.2 VDF from supersingular curves over $\mathbb{F}_{p^2}$

Our second VDF is very similar to the previous one, but uses supersingular curves defined over $\mathbb{F}_{p^2}$, thus sharing some similarities with the Charles–Goren–Lauter hash function [14], and with SIDH [33,23]. It deviates slightly from the paradigm presented in Section 3 in that the inputs are not taken in a cyclic group, and evaluation is slower than the previous one by a factor of about 2 (see Section 7), but has some advantages over it that will be discussed in Section 6.

Like before, we choose a prime p such that $p+1$ contains a large prime factor N, and a small prime ℓ, e.g., $\ell = 2$.[12] We again choose a supersingular elliptic curve $E/\mathbb{F}_p$ (this will be necessary to define the orthogonal groups X_1, X_2), however we see it as a curve over $\mathbb{F}_{p^2}$, so that $t = -2p$ and $\#E(\mathbb{F}_{p^2}) = (p+1)^2$.

Like before, we define $X_1 = \upsilon^{-1}(\tilde{E}[N] \cap \tilde{E}(\mathbb{F}_p))$ and $X_2 = E[N] \cap E(\mathbb{F}_p)$, where $\tilde{E} = \upsilon(E)$ is a quadratic twist of E over $\mathbb{F}_p$. The maps $\phi, \hat{\phi}$ are again a cyclic isogeny of degree ℓ^T and its dual, however, over $\mathbb{F}_{p^2}$, there are $(\ell+1)\ell^{T-1}$ possible choices for them, instead of just two; we will select one of them by doing a non-backtracking random walk in the full ℓ-isogeny graph.

On the image curve E', we define $Y_1 = \phi(X_1)$ and $Y_2 = \phi(X_2)$. However, we are now faced with a difficulty: there is no known efficient way to sample from Y_2 or Y_1, indeed E' is generally defined over $\mathbb{F}_{p^2}$ and it has therefore no $\mathbb{F}_p$-twists. To bypass this problem, we deviate from the abstract description of Section 3, obtaining an N-to-1 map instead of a bijection. Let π be the Frobenius endomorphism of $E/\mathbb{F}_p$, the *trace map* on $E/\mathbb{F}_{p^2}$ is the map

$$\begin{aligned} \mathrm{Tr} : E/\mathbb{F}_{p^2} &\to E/\mathbb{F}_p, \\ P &\mapsto P + \pi(P). \end{aligned}$$

[12] For this VDF, there is no practical reason to choose any other prime than $\ell = 2$.

$\mathsf{Setup}(\lambda, T)$

1. Choose primes N, p with the properties above, according to the security parameter λ;
2. Select a supersingular curve $E/\mathbb{F}_p$;
3. Perform a random non-backtracking walk of length T in the ℓ-isogeny $\mathbb{F}_{p^2}$-graph, defining a cyclic ℓ^T-isogeny $\phi : E \to E'$ and its dual $\hat{\phi}$;
4. Choose a generator P of $X_1 = \upsilon^{-1}(\tilde{E}[N] \cap \tilde{E}(\mathbb{F}_p))$, and compute $\phi(P)$;
5. Output $(\mathsf{ek}, \mathsf{vk}) = \big(\hat{\phi}, (E, E', P, \phi(P))\big)$.

$\mathsf{Eval}(\hat{\phi}, Q \in E'[N])$

1. Compute and output $(\mathrm{Tr} \circ \hat{\phi})(Q)$.

$\mathsf{Verify}(E, E', P, Q, \phi(P), (\mathrm{Tr} \circ \hat{\phi})(Q))$

1. Verify $(\mathrm{Tr} \circ \hat{\phi})(Q) \in X_2 = E[N] \cap E(\mathbb{F}_p)$;
2. Verify that $e_N(P, (\mathrm{Tr} \circ \hat{\phi})(Q)) = e_N(\phi(P), Q)^2$.

Fig. 2. Instantiation of the Verifiable Delay Function over $\mathbb{F}_{p^2}$

In particular, the trace map sends $E[N]$ to X_2, and satisfies

$$e_N(P, \mathrm{Tr}(R)) = e_N(P, (1+\pi)(R)) = e_N((1-\pi)(P), R) = e_N([2]P, R) = e_N(P, R)^2$$

for all $P \in X_1$ and $R \in E[N]$. We thus define our VDF as

$$\begin{aligned} f : E'[N] &\to X_2, \\ Q &\mapsto (\mathrm{Tr} \circ \hat{\phi})(Q); \end{aligned}$$

verification is done by checking a pairing equation as before. The algorithms are described in Figure 2.

A bijective VDF over $\mathbb{F}_{p^2}$. If a bijection is wanted, an alternative VDF using the $\mathbb{F}_{p^2}$-graph would swap roles by having E' defined over $\mathbb{F}_p$, and E over $\mathbb{F}_{p^2}$. During the Setup phase, a basis (P, R) of $X_1 \times X_2$ is sampled by evaluating $\hat{\phi}/2^T$ on a basis of $Y_1 \times Y_2$, and it is added to the verification key vk. Then, sampling Q in Y_2 is easy, and verifying that $\hat{\phi}(Q) \in X_2$ can be done by checking that $e_N(R, \hat{\phi}(Q)) = 1$. However this protocol is less efficient, because verification requires two pairing computations.

5.3 Properties of the VDFs.

In slight disagreement with the definitions of [7], the Setup routines presented here take $O(T)$ time to compute the isogenies $\phi, \hat{\phi}$, and produce evaluation keys of size $O(T)$. While the size of the evaluation key can be reduced by redoing parts of the computation in Eval (see Section 7), the only known way to verify the public parameters is to, essentially, rerun the Setup.

We also note that, although T can be arbitrary (we discuss bounds on T in the next section), neither of our VDFs is *incremental* in the sense of [7], meaning

that a single parameter set produced by Setup shall support more than one delay T. A possible workaround is to have Setup include some intermediate curves in the verification key, so that a single Setup can be used for many delay parameters up to T, at the cost of increasing the size of the verification key.

Finally, the VDF over $\mathbb{F}_p$ is *decodable* in the sense of [7], meaning that given the output $\hat{\phi}(Q)$ one can compute the input Q (although not more efficiently than evaluating $\hat{\phi}$); the VDF over $\mathbb{F}_{p^2}$, on the other hand, is obviously not decodable because it is non-injective.

6 Security and parameter sizes

We now give formal security definitions and proofs, following [7]. A VDF must satisfy three security properties: *correctness*, *soundess*, and *sequentiality*, as defined below. In [7], soundness is a weaker property where the evaluator is allowed a negligible cheating probability; we introduce here the stronger notion of unconditional soundness, which is achieved by our VDFs.

Definition 1 (Correctness, soundness). *The VDFs of Section 5 are* correct *if, for any λ, T, public parameters $(\mathsf{ek}, \mathsf{vk}) \leftarrow \mathsf{Setup}(\lambda, T)$, and all input Q, if $R \leftarrow \mathsf{Eval}(\mathsf{ek}, Q)$ then $\mathsf{Verify}(\mathsf{vk}, Q, R)$ outputs* true.

They are unconditionally sound *if for all λ, T, public parameters $(\mathsf{ek}, \mathsf{vk}) \leftarrow \mathsf{Setup}(\lambda, T)$, and all input Q, if $R \neq \mathsf{Eval}(\mathsf{ek}, Q)$ then $\mathsf{Verify}(\mathsf{vk}, Q, R)$ outputs* false.

Theorem 1. *The VDFs of Section 5 are correct and unconditionally sound.*

Proof. The map $R \mapsto e_N(P, R)$ is a group isomorphism between the output space $X_2 \subset E[N]$ and the multiplicative subgroup $\mu_N \subset \mathbb{F}_{p^2}$. Hence, verification succeeds if and only if the output is correct.

Sequentiality is the defining property of VDFs, and is much subtler to define. Intuitively, we want it to be impossible to evaluate the VDF *faster* than running Eval, even given an unbounded amount of parallel resources, and even if the adversary is allowed a *large* amount of precomputation after the public parameters are generated. We must of course exclude trivial cases where, for example, the adversary precomputes a list of input-output pairs, hence we model security as a game where the adversary is allowed a polynomial amount of precomputation, after which he receives a random input point Q and must produce the output $\hat{\phi}(Q)$ (or $\mathrm{Tr} \circ \hat{\phi}(Q)$) faster than Eval with non-negligible probability. We also introduce here a new definition: if the adversary cannot break sequentiality, even when he is allowed a quantum precomputation before seeing the point Q, we say that the VDF is quantum annoying.

Definition 2 (Sequentiality, quantum annoyance). *The VDFs of Section 5 are* sequential *if no pair of randomized algorithms $\mathcal{A}_0$, which runs in total time* $\mathrm{poly}(T, \lambda)$*, and $\mathcal{A}_1$, which runs in parallel time less than T, can win with non-negligible probability the following sequentiality game*

1. $(\mathsf{ek}, \mathsf{vk}) \xleftarrow{\$} \mathsf{Setup}(\lambda, T)$, *where all randomness in* Setup *is generated by fair coin tosses,*
2. $A \leftarrow \mathcal{A}_0(\lambda, \mathsf{ek}, \mathsf{vk}, T)$,
3. $Q \xleftarrow{\$} Y_2$, *uniformly sampled,*
4. $Q' \leftarrow \mathcal{A}_1(A, \mathsf{vk}, Q)$,

where winning is defined as outputting $Q' = \hat{\phi}(Q)$ *(or* $Q' = \mathrm{Tr} \circ \hat{\phi}(Q)$*).*

Moreover, if $\mathcal{A}_0$ *is allowed a quantum computation in* $\mathrm{poly}(T, \lambda)$*, we say that the VDFs are* quantum annoying.

We leave aside the question of formally defining a computational model where "running in parallel time less than T" has a definite meaning; see [7,64] for details.

We shall see soon that Setup must use secret randomness to select the starting curve $E/\mathbb{F}_p$; after that, Setup is only left with choosing the isogeny $\phi : E \to E'$ and the generator $P \in E[n]$, and both choices can done using public randomness. Furthermore $\mathcal{A}_0$ is allowed $\mathrm{poly}(T)$ computation, so it can compute $\hat{\phi}$ and evaluate ϕ on P (and also evaluate $\hat{\phi}$ on polynomially many points of Y_2). Hence, choice of $E/\mathbb{F}_p$ aside, Setup can be absorbed into $\mathcal{A}_0$; this justifies defining the following problem, which is a simple rewording of the sequentiality hypothesis:

Definition 3 (Isogeny shortcut problem (over k)). *Let* $E/\mathbb{F}_p$ *be a curve uniformly sampled in the set of all supersingular curves defined over a finite field* $\mathbb{F}_p$*. Given an isogeny* $\phi : E \to E'$ *of degree* ℓ^T *to a curve* E'/k*, with* $k = \mathbb{F}_p$ *or* $k = \mathbb{F}_{p^2}$*; being allowed a precomputation taking total time* $\mathrm{poly}(T, \lambda)$*, evaluate* $\hat{\phi}(Q)$ *on a random point* $Q \in E'[N] \cap E'(k)$ *in parallel time less than* T*.*

6.1 Attacks

We now discuss some attacks on the isogeny shortcut problem, and use them to set parameter sizes.

Pairing inversion. The simplest attack exploits the same properties as the verification. It works both against the VDFs and the generalization of BLS signatures sketched in Section 3. Given $P, \phi(P), Q$, to compute $\hat{\phi}(Q)$ (or $(\mathrm{Tr} \circ \hat{\phi})(Q)$) it is enough to solve the *pairing inversion problem* $e_N(P, \cdot\,) = e_N(\phi(P), Q)$. Note that this attack must be repeated for each new input Q.

The hardness of the pairing inversion problem impacts the size of N and p. Given that our curves have embedding degrees 2 or 1, the best algorithm at our disposal is the Number Field Sieve for $\mathbb{F}_{p^2}$, with (heuristic) complexity $L_p(1/3)$. The current record for computing discrete logarithms in $\mathbb{F}_{p^2}$ is for a prime p of almost 300 bits [2], while for a security of 128 bits it is recommended to take p of around 1500 bits, and N of 256 bits.

Computing shortcuts. A different path to breaking our VDFs consists in finding a "simpler" isogeny from E to E', agreeing with ϕ on $E[N]$, but taking less parallel time to compute. This kind of attacks can be decomposed in two steps: first find a "simpler" isogeny $\psi : E \to E'$ (e.g., of lower degree), then find an endomorphism $\omega \in \mathrm{End}(E)$ such that $\omega \circ \hat{\psi}$ agrees with $\hat{\phi}$ on $E'[N]$.

Note that when $\deg \phi$ is super-polynomial in p a lower degree isogeny $\psi : E \to E'$ always exists; indeed Pizer [51,52] has shown that ℓ-isogeny graphs of supersingular curves over $\bar{\mathbb{F}}_p$ are optimal expanders for any prime ℓ, and thus have diameter in $O(\log(p))$, implying that there is an ℓ-isogeny walk connecting E to E' of degree polynomial in p. However, it may be difficult to compute such an isogeny in general: the best generic algorithm in the case of $\mathbb{F}_{p^2}$-graphs is a birthday paradox method [30,24], that finds a collision in $O(\sqrt{p})$ isogeny steps on average; even then, the resulting isogeny has degree much larger than wanted. Note that the only quantum speedup known for this problem is a generic Grover search, giving a square-root acceleration at best [5].

For curves over $\mathbb{F}_p$, computing the structure of the class group $\mathrm{Cl}(\mathrm{End}(E))$ allows an attacker to find an equivalent isogeny ψ, potentially with lower degree. As a very basic example, by computing the class group structure the attacker may learn that the ideals associated to horizontal ℓ-isogenies have order $t < T$, and thus that there is an isogeny of degree $\ell^{T \bmod t}$ from E to E'. This attack is essentially equivalent to computing the group order in Wesolowski's VDF. For a more comprehensive attack, the algorithm of Jao and Soukharev [35] allows an attacker to compute an equivalent ideal of smooth norm using $L_p(1/2)$ operations. We will sketch later how this same task can be achieved in polynomial time on a quantum computer.

After computing a "simpler" isogeny $\psi : E \to E'$, we are left with the problem of finding ω such that $\omega \circ \hat{\psi} = \hat{\phi}$ on $E'[N]$. This problem can be solved by computing discrete logarithms in $E[N]$, which is not easier than the pairing inversion problem mentioned above; however, this attack needs only be performed once on the public parameters, and can then be used to speed up any evaluation.

So far, we have only discussed the computation of *shortcuts* in the generic case; however, when E or E' are special curves, there are much better ways to solve this problem, that would lead to a complete break of our VDFs. We discuss this issue in Subsection 6.2.

Parallel isogeny evaluation. Finally, the last attack path would be to find a better (parallel) algorithm for evaluating isogenies of degree ℓ^T. All known algorithms require to go through each of the T intermediate curves, one after the other. It is certainly possible to aggregate steps in blocks, e.g., replace two 2-isogenies with one 4-isogeny, as it is typically done in implementations of SIDH/SIKE [19], however the complexity of each block grows exponentially with the block length, thus it is not clear how to use parallelism to evaluate each block faster. This is certainly the newest and most unusual problem in the area of elliptic curve cryptography, and the one that needs more investigation. Note that progress in this direction could eventually benefit other isogeny-based cryptographic protocols, such as key exchange [33,13,1] and signatures [65,22].

Bounds on T. None of the attacks so far has set an upper bound on T. By the birthday paradox, we shall take T smaller than the square root of the size of the isogeny graph, because a loop in the isogeny walk could be optimized away from Eval. Given that the size of the isogeny graphs is $O(\sqrt{p})$ and $O(p)$ respectively, we obtain bounds of $O(\sqrt[4]{p})$ for $\mathbb{F}_p$, and $O(\sqrt{p})$ for $\mathbb{F}_{p^2}$.

However, these bounds are much higher than the best attacks, that are subexponential in p. Thus T is effectively only bounded by the theoretical limit of being subexponential in λ.

Quantum security. We briefly analyze our proposals in the post-quantum setting. Obviously, Shor's algorithm breaks the pairing inversion problems in polynomial time, thus our VDFs cannot be considered post-quantum. However, looking at Definition 2, we see that this attack can only be applied after the input point Q is given to $\mathcal{A}_1$; thus our VDFs have a chance of being *quantum annoying* as defined there. In a plausible future where quantum computers do exist, but are very expensive and slow, it may still be more interesting to evaluate the VDF in the legitimate way, rather than attack the pairing inversion problem with Shor's algorithm. We argue that, given current knowledge, our VDF over $\mathbb{F}_{p^2}$ is quantum annoying, whereas the one over $\mathbb{F}_p$ is not.

Indeed, as long as the input point Q is unknown, the only strategy currently available $\mathcal{A}_0$ is to compute an isogeny shortcut, as described previously. In the $\mathbb{F}_{p^2}$ case, this would involve finding a cycle in the isogeny graph through $E/\mathbb{F}_p$ and $E'/\mathbb{F}_{p^2}$, a problem that is believed to be quantum-resistant when E and E' are generic supersingular curves [31,27].

For the $\mathbb{F}_p$ case, on the other hand, it is enough to compute the structure of $\mathrm{Cl}(\mathrm{End}(E))$, along with a basis of "short" generators, a task doable in polynomial time on a quantum computer using Kitaev's generalization of Shor's algorithm [37]; then an isogeny $\psi : E \to E'$ of smooth degree defined over $\mathbb{F}_p$ can be computed efficiently on a classical computer by solving a closest vector problem (see [22, Appendix C] for a similar algorithm). Finally, since ψ and ϕ are both defined over $\mathbb{F}_p$, the subgroup $X_2 = E[N] \cap E(\mathbb{F}_p)$ is an eigenspace for the endomorphism $\hat{\psi} \circ \phi$; then a discrete logarithm computation in X_2 finds a scalar s such that $[s] \circ \hat{\psi} = \hat{\phi}$ on Y_2.

Security of the identification protocol. For completeness, we briefly come back to the security of the generalization of the BLS identification protocol sketched in Section 3.

We are not interested in sequentiality in this case, thus shortcut attacks are not relevant here. Instead, key recovery is equivalent to the problem of finding a secret isogeny $\phi : E \to E'$, given E, E', a basis (P, Q) of $E[N]$, and $\phi(P), \phi(Q)$. This problem is much more similar to classical problems in isogeny based cryptography, and is obviously harder than the isogeny shortcut problem.

The best known classical attacks, both for the $\mathbb{F}_p$ and the $\mathbb{F}_{p^2}$ case, are in the square root of the graph size (respectively, $O(\sqrt[4]{p})$ and $O(\sqrt{p})$). But key recovery is hard even for quantum computers: the best attack for the $\mathbb{F}_p$ case is Kuperberg's algorithm for the *Hidden Shift Problem* [41,54,42,15,10,4,34,3], which

finds ϕ in $\exp(\sqrt{\log(p)})$ quantum operations; whereas in the $\mathbb{F}_{p^2}$ case quantum computers give a square-root speedup via Grover's algorithm at best [5].

Hence, both identification protocols have a security property similar to the *quantum annoyance* defined in Defintion 2: any forgery requires running a new instance of Shor's algorithm, while key recovery is infeasible on quantum computers. This may be a useful replacement for basic BLS signatures in contexts where Shor's algorithm is slow and expensive, and signatures must be produced fast.

Finally, we remark that our protocol, unlike BLS, is *succinct*, in the sense that the secret isogeny is potentially sub-exponentially larger than the proof of knowledge. At present, this seems rather limited, since our protocol is not zero-knowledge, however we hope that further research may add more useful properties to it.

6.2 Shortcut attacks on special curves

We now come back to the *shortcut* attacks analyzed previously. We saw that the best algorithms available in the general case have exponential or sub-exponential complexity, and are *in general* not better than a simple pairing inversion attack. However, when the endomorphism ring of the starting curve E is known, a much better algorithm exists, completely breaking sequentiality of our VDFs. We now present a sketch of the attack, and the only known solution to avoid it.

Attack overview. We shall suppose that the delay parameter T is super-linear in $\log(p)$. To simplify our description we also assume that E is the curve defined by the equation $y^2 = x^3 + x$, with j-invariant $j = 1728$. However, the attack can be generalized to an arbitrary curve provided we know its endomorphism ring. It can also be applied to our VDF over $\mathbb{F}_p$, because an attacker is not bound to keep all computations in $\mathbb{F}_p$.

The attack has two main steps. First, we compute an alternative isogeny $\psi : E \to E'$ with a powersmooth and reasonably small degree (polynomial in p). This is achieved by adapting a strategy used in [48,27] to compute a collision to Charles–Goren–Lauter (CGL) hash function [14]. Second, we compute an endomorphism $\omega \in \mathrm{End}(E)$ such that the actions of $\omega \circ \hat{\psi}$ and $\hat{\phi}$ are identical on $E[N]$. By expressing ω on a set of generators of $\mathrm{End}(E)$, we are able to evaluate $\omega \circ \hat{\psi}$ efficiently on $E[N]$, and thus we can answer evaluation queries in a time much shorter than T.

Computing shortcuts. Let $\phi : E = E_0 \to E_T = E'$ be given as a composition of degree ℓ isogenies. We now show how to compute an alternative isogeny $\psi : E \to E'$ with much shorter degree.

A natural idea to solve this problem is to translate this problem to an analogous problem in the quaternion algebra $B_{p,\infty}$ ramified at p and at infinity, solve the problem in the quaternion algebra, and translate the solution back to the geometric setting. Indeed $\mathrm{End}(E_0)$ is isomorphic to a maximal order $\mathcal{O}_0$ of

$B_{p,\infty}$, and by assumption on E this isomorphism is fully known. Translating the problem back and forth (from isogenies to their corresponding ideals and conversely) can be done using techniques dating back to Waterhouse [63], and the "quaternion isogeny" algorithm of Kohel, Lauter, Petit and Tignol (KLPT) [39] can be used to solve the problem in the quaternion algebra. Unfortunately, the translation algorithms require to compute torsion points of order $\deg \phi$, which have exponential size in general.

We adapt an idea used in the collision algorithm of [48,27] to avoid this problem. Let $\phi_i : E_0 \to E_i$ correspond to the first i steps of the isogeny. Let I_i be the corresponding ideal, and let $n(I_i) = \ell^i$ denote its norm. Assume we have already computed an ideal J_i in the class of I_i with powersmooth norm. We sketch how to compute an ideal in the class of I_{i+1} with powersmooth norm.

1. Compute the $\ell + 1$ ideals $K_{i+1,k}$, $k = 0, \ldots, \ell$ with norm $n(J_i)\ell$ such that $K_{i+1,k} \bmod n(J_i)\mathcal{O}_0 = J_i$ (algorithms for this task are provided in [36]).
2. Apply the powersmooth quaternion isogeny algorithm to each $K_{i+1,k}$ to obtain new ideals $J_{i+1,k}$ in the same classes respectively.
3. Translate each ideal $J_{i+1,k}$ to an isogeny $\psi_{i+1,k}$.
4. Identify the (usually unique) k such that the image of $\psi_{i+1,k}$ has j-invariant $j_{i+1} = j(E_{i+1})$.
5. Let $J_{i+1} = J_{i+1,k}$.

To obtain the desired isogeny ψ, we repeat those steps for $i = 1, \ldots, T-1$. When $i = T - 1$, we additionally set $\psi = \psi_{T,k}$.

The heuristic bounds and the experiments in [39] show that the degree of ψ_i is polynomial in p (more precisely $O(p^{7/2})$) and the computation can be completed in time $\mathrm{poly}(T, \log(p))$. The isogeny $\psi : E \to E'$ has powersmooth degree much smaller than that of ϕ, and can therefore be evaluated much faster.

Matching image points on the N-torsion. Let $\iota : E \to E : (x, y) \to (-x, iy)$ where $i^2 = -1$, and let $\pi : E \to E : (x, y) \to (x^p, y^p)$. We have $\mathrm{End}(E) = \langle 1, \iota, \frac{1+\pi}{2}, \frac{\iota+\pi\iota}{2} \rangle$, so the endomorphism $\theta := \hat{\psi} \circ \phi$ can be written as $\theta = a_0 + a_1\iota + a_2\pi + a_3\pi\iota$ with $a_0, a_1, a_2, a_3 \in \mathbb{Z}[1/2]$. Moreover we have

$$a_i = \langle \theta, \alpha_i \rangle := (\theta \circ \hat{\alpha}_i + \alpha_i \circ \hat{\theta})/2$$

for $\alpha_i = 1, \iota, \pi, \pi\iota$ respectively, and these coefficients can be computed using a variant of Schoof's algorithm [38, Theorem 81], by evaluating those maps on small torsion points and applying the Chinese remainder theorem. Note that $|a_i| \leq \deg\theta \deg\alpha_i$, so this computation can be performed in time $\mathrm{poly}(T, \log p)$.

If we now set $\omega = r\hat{\theta}$, where $r = \ell^T/(a_0^2 + a_1^2 + pa_2^2 + pa_3^2)$, then $\omega \circ \hat{\psi} = \hat{\phi}$. But ω can be evaluated at any point of E as

$$\omega(Q) = \sum [ra_i \bmod N]\, \hat{\alpha}_i(Q),$$

at a cost of only $O(\log(N))$ operations. Thus we can replace $\mathsf{Eval}(Q)$ with the evaluation of $\hat{\psi}$ followed by the evaluation of ω, for a total costs of only $\mathrm{polylog}(p)$, which is less than T by hypothesis.

Countering the attack. The KLPT algorithm only works when the starting curve E has an endomorphism ring that is, in their words, *extremal* and *special*. *Extremal* means that E is defined over $\mathbb{F}_p$, a condition common to all instantiations of our VDF; however only few curves are also *special*, for example $j(E) = 1728$, or other curves with complex multiplication by an oder with small discriminant.

The KPLT algorithm extends to a non-special curve E, when a path $E \to E_0$ to a special curve E_0 is known. Unfortunately, all known methods to select random supersingular curves do so by starting a random walk from some special curve; hence, there is no known way to produce a random supersingular curve $E/\mathbb{F}_p$ without producing a backdoor $E \to E_0$.

At present, the only way to counter the attack presented here is to use a trusted setup to produce a random curve $E/\mathbb{F}_p$, i.e., having a trusted authority (or many trusted authorities engaged in a multi-party protocol) compute a walk $E_0 \to E$ from a special curve E_0, and then throw the backdoor away.

A note on ordinary curves. It is natural to ask whether it is possible to obtain VDFs from ordinary isogeny graphs. Although it is conceivable to have a variant of our VDF over $\mathbb{F}_p$ using ordinary curves, no secure instantiation is currently known. Indeed, all known ordinary pairing-friendly curves are obtained using variations of the CM method, and thus have small quadratic discriminant and small isogeny class. In this case it is possible to compute the structure of $\mathrm{End}(E)$, and do a shortcut attack similar to the one above.

We proceed in two steps as before. We first find an isogeny $\psi : E \to E'$ of small powersmooth degree; since the isogeny class is small, this can even be done by exhaustive search.

Then, we are left with the problem of finding $\omega \in \mathrm{End}(E)$ such that $\omega \circ \hat{\psi} = \hat{\phi}$ when restricted to $E[N]$. We proceed as before: using Schoof's algorithm we compute $\theta = \hat{\psi} \circ \phi = a + b\pi$ for some $a, b \in \mathbb{Q}$, then we set $\omega = \ell^T \hat{\theta}/(a^2 + pb^2)$, and we replace Eval by $\omega \circ \hat{\psi}$.

A family of ordinary pairing friendly elliptic curves with generic discriminant would provide the perfect instantiation for our VDFs, as it would not be vulnerable to any known shortcut attack, and thus would not need a trusted setup. Unfortunately, all known constructions of pairing-friendly elliptic curves use complex multiplication hence produce curves with small discriminants [28].

7 Implementation

Our proposed VDFs can be easily implemented using the fundamental blocks already available for pairing-based and isogeny-based cryptography. In particular, our curves are typical supersingular pairing-friendly ones with no additional properties, thus they are likely to be supported in most specialized software libraries for pairings.

A drawback of our method being the long setup time and the large evaluation key, we present here an implementation that improves both by orders of magnitude.

We focus on 2-isogenies, as they are the most obvious candidate for an implementation. There are two standard ways to compute a 2-isogeny walk from a curve $E : y^2 = f(x)$. The first is to factor the 2-division polynomial $f(x)$ to obtain all the points of order 2, then use Vélu's formulas [59] to test all directions and step in the wanted one. Since Vélu's formulas also produce the generator of the dual isogeny to the direction one is coming from, this root can be quotiented out from $f(x)$, and thus we are left with solving one square root per curve. The second way is to take a point at random on E and multiply it by the cofactor $\#E/2$. If we obtain a 2-torsion point defining the wanted direction, then we compute it and we move to the next curve; otherwise we try with a different point. Both ways require $O(\log(p))$ operations in the base field for one step, and thus $O(T \log(p))$ operations to compute the full isogeny walk. After the isogeny ϕ is computed, the list of the kernel points can be stored so to be able to evaluate ϕ in $O(T)$ operations. However, this implies storing T points and curves, which may require a large storage.

Fortunately, using isogeny evaluation techniques pioneered in SIDH [23], and applied in [25] to the CGL hash function [14], it is possible to absorb the $\log(p)$ factor and shorten the evaluation key size by the same amount. For this, we choose a prime of the form $p = 2^n fN - 1$, so that all curves in the isogeny graph have rational points of order 2^{n-1} or 2^n (depending on whether we use $\mathbb{F}_p$-graphs or $\mathbb{F}_{p^2}$-graphs). This way, a single point P_i on E_i can be used to define n (or $n-1$) consecutive steps in the graph, and the corresponding isogeny can be evaluated in $n \log(n)$ operations using the *optimal strategy* techniques from [23].

More in detail, in the $\mathbb{F}_{p^2}$ case, the curve E_i has group structure $E(\mathbb{F}_{p^2}) \simeq (\mathbb{Z}/(p+1)\mathbb{Z})^2$, and is usually not defined over $\mathbb{F}_p$. We can compute a point P_i of order 2^n by taking a point at random and multiplying by fN, then verifying that P_i has the wanted order. We check that P_i does not start a backtracking isogeny and we use it to advance n steps in the graph, then we start again.

In the $\mathbb{F}_p$ case, because we chose a curve on the surface, the group structure is $E(\mathbb{F}_p) \simeq \mathbb{Z}/\frac{p+1}{2}\mathbb{Z} \times \mathbb{Z}/2\mathbb{Z}$ (see [45]), hence the highest order we can get for P_i is 2^{n-1}. Such point P_i will define 2^{n-2} horizontal isogeny steps in the "positive" direction determined by the ideal $(\pi - 1) \subset \mathrm{End}(E)$, plus one last step that is either in the same direction, or going to the floor. To avoid "getting stuck" on the floor, we use P_i to advance $n-2$ steps, then start again.

Using these techniques, only $\approx T/n$ points need to be computed and the full walk is computed in $O(T \log n)$ operations. One has the choice between storing all the intermediate 2-torsion points, or storing only the higher order points P_i. In the first case, we use $O(T)$ storage and evaluation time; in the second case, we use $O(T/n)$ storage and $O(T \log n)$ evaluation time. Since $n \approx \log p$, the slowdown in the second case is likely to be negligible in front of other factors, such as data transfer delays, or speedups due to dedicated hardware.

In practice, we use a projective (x, z)-only Montgomery model for our curves, for which small degree isogeny formulas are the most efficient [19]. Points defined over $\mathbb{F}_{p^2}$ are then stored in $4 \log_2(p)$ bits, and a curve is represented by $y^2 = x^3 + ax^2 + x$ using $2 \log_2(p)$ bits. The isogeny $\hat{\phi}$ is decomposed in

small degree isogenies, and each one is represented by its kernel and its image curve. If we choose to represent $\hat{\phi}$ as a composition of 2-isogenies, its representation is stored in $2T\log_2(p)$ bits. If we decide to represent it as a composition of 2^n-isogenies, storing kernels and curve coefficients requires $T/n(4\log_2(p) + 2\log_2(p)) = 6T\log_2(p)/n$ bits.

For verification, we use the Tate pairing that we decompose in two parts: the Miller loop and the exponentiation. Because of the special prime p, the final exponentiation is very efficient. Indeed,

$$\frac{p^2-1}{N} = (p-1)\frac{p+1}{N} = (p-1)2^n f = (2^n fN - 2)2^n f = 2^{n+1} f(2^{n-1} fN - 1)$$

Concretely, for a 128-bit secure VDF, we choose a prime N of 256 bits

N=86910174218680023830821387559262103190768292159253431612727227446130211466083,

and a prime p of 1503 bits defined by $p = 2^{1244}63N - 1$.

To validate our proposed parameters, we implemented a (non-optimized) proof of concept in SageMath [58]. For a delay parameter $T \approx 2^{16}$, we obtain the time and memory costs given in Table 2.

Protocol	**Step**	**Storage size**	**Time**
$\mathbb{F}_p$	Setup	238 kb	1416 s
	Evaluation	–	2056 s
	Verification	–	7 s
$\mathbb{F}_{p^2}$	Setup	491kb	2727 s
	Evaluation	–	2817 s
	Verification	–	7 s

Table 2. Benchmarks and storage estimations for our VDFs, on a Intel Xeon E5-2609 @ 2.40GHz, on SageMath 8.5

8 Conclusion and perspectives

We presented two new candidate Verifiable Delay Functions, based on assumptions from pairing-based and isogeny-based cryptography. Our VDFs are practical, and offer several advantages over previous proposals.

At present, our constructions require a trusted setup to generate initial parameters. It is an important open problem to find an algorithm to generate random supersingular curves in a way that does not reveal their endomorphism ring, and we encourage the community to work on it. As long as such an algorithm is missing, it is interesting to look for efficient multi-party algorithms for doing isogeny walks.

It would also be interesting to reduce the cost of validating public parameters, ideally to a time independent from the delay parameter T. Relatedly, our

VDFs have large storage requirements for the evaluator; in our implementation we presented a way to mitigate this issue, however this creates a compromise between storage and evaluation time, that needs to be carefully considered by the evaluator, depending on the intended application. More research on practical ways to mitigate the price of the large storage is desirable.

Here we only sketched the shortcut attack against insecure instances using special curves. It would be interesting to do a more detailed analysis of its complexity, of its limitations, and of its possible generalizations; we leave this as future work. We also encourage research on alternative ways to break the Isogeny Shortcut Problem, for example finding ways to parallelize isogeny evaluation.

Finally, our VDFs can be seen as a generalization of BLS signatures: if the isogeny is kept secret, we obtain a proof of knowledge of an isogeny walk between two curves, that can be used for identification or signatures. At the moment, the only advantage over BLS signatures is a weak form of quantum resistance; we hope that further research would add useful properties to our protocol enabling more applications.

Acknowledgments. We would like to thank Bill Allombert, Razvan Barbulescu, Jeff Burdges, Wouter Castryck, Jeroen Demeyer, Andreas Enge, Steven Galbraith, Matthew Green, Jean Kieffer, Enea Milio, Aurel Page, Damien Robert, Barak Shani and Benjamin Wesolowski for fruitful discussions.

Luca De Feo was supported by the French *Programme d'Investissements d'Avenir* under the national project RISQ n° P141580-3069086/DOS0044212.

References

1. Azarderakhsh, R., Koziel, B., Campagna, M., LaMacchia, B., Costello, C., Longa, P., De Feo, L., Naehrig, M., Hess, B., Renes, J., Jalali, A., Soukharev, V., Jao, D., Urbanik, D.: Supersingular isogeny key encapsulation (2017), http://sike.org
2. Barbulescu, R., Gaudry, P., Guillevic, A., Morain, F.: Improving nfs for the discrete logarithm problem in non-prime finite fields. In: Oswald, E., Fischlin, M. (eds.) Advances in Cryptology – EUROCRYPT 2015. pp. 129–155. Springer Berlin Heidelberg, Berlin, Heidelberg (2015)
3. Bernstein, D.J., Lange, T., Martindale, C., Panny, L.: Quantum circuits for the CSIDH: optimizing quantum evaluation of isogenies. Cryptology ePrint Archive, Report 2018/1059 (2018), https://eprint.iacr.org/2018/1059
4. Biasse, J.F., Iezzi, A., Jacobson, M.J.J.: A note on the security of CSIDH. In: Kangacrypt 2018 (2018), https://arxiv.org/abs/1806.03656
5. Biasse, J.F., Jao, D., Sankar, A.: A quantum algorithm for computing isogenies between supersingular elliptic curves. In: International Conference in Cryptology in India. pp. 428–442. Springer (2014)
6. Blake, I.F., Seroussi, G., Smart, N., et al.: Advances in Elliptic Curve Cryptography, London Mathematical Society Lecture Note Series, vol. 317. Cambridge University Press, New York, NY, USA (2005)
7. Boneh, D., Bonneau, J., Bünz, B., Fisch, B.: Verifiable delay functions. In: Shacham, H., Boldyreva, A. (eds.) Advances in Cryptology – CRYPTO 2018. pp. 757–788. Springer International Publishing, Cham (2018)

8. Boneh, D., Bünz, B., Fisch, B.: A survey of two verifiable delay functions. Cryptology ePrint Archive, Report 2018/712 (2018), https://eprint.iacr.org/2018/712
9. Boneh, D., Lynn, B., Shacham, H.: Short signatures from the Weil pairing. Journal of Cryptology **17**(4), 297–319 (Sep 2004). https://doi.org/10.1007/s00145-004-0314-9
10. Bonnetain, X., Schrottenloher, A.: Quantum security analysis of CSIDH and ordinary isogeny-based schemes. Cryptology ePrint Archive, Report 2018/537 (2018), https://eprint.iacr.org/2018/537
11. Broker, R.M., Charles, D.X., Lauter, K.E.: Cryptographic applications of efficiently evaluating large degree isogenies (Aug 2012), US Patent 8,250,367
12. Buchmann, J., Hamdy, S.: A survey on iq cryptography. In: In Proceedings of Public Key Cryptography and Computational Number Theory. pp. 1–15 (2001)
13. Castryck, W., Lange, T., Martindale, C., Panny, L., Renes, J.: CSIDH: An efficient post-quantum commutative group action. In: Peyrin, T., Galbraith, S.D. (eds.) Advances in Cryptology – ASIACRYPT 2018. pp. 395–427. Springer International Publishing (2018)
14. Charles, D.X., Goren, E.Z., Lauter, K.E.: Cryptographic hash functions from expander graphs. Journal of Cryptology **22**(1), 93–113 (Jan 2009). https://doi.org/10.1007/s00145-007-9002-x
15. Childs, A., Jao, D., Soukharev, V.: Constructing elliptic curve isogenies in quantum subexponential time. Journal of Mathematical Cryptology **8**(1), 1–29 (2014)
16. Cohen, B.: Proofs of space and time. Blockchain Protocol Analysis and Security Engineering (2017), https://cyber.stanford.edu/sites/default/files/bramcohen.pdf
17. Cohen, B.: Lipa long. Chia Network (2018), https://github.com/Chia-Network/vdf-competition/blob/master/classgroups.pdf
18. Cohen, B., Pietrzak, K.: Simple proofs of sequential work. Cryptology ePrint Archive, Report 2018/183 (2018), https://eprint.iacr.org/2018/183.pdf
19. Costello, C., Longa, P., Naehrig, M.: Efficient algorithms for Supersingular Isogeny Diffie-Hellman. In: Robshaw, M., Katz, J. (eds.) Advances in Cryptology – CRYPTO 2016: 36th Annual International Cryptology Conference. pp. 572–601. Springer Berlin Heidelberg (2016). https://doi.org/10.1007/978-3-662-53018-4_21
20. Cox, D.A.: Primes of the form x2 + ny2: Fermat, class field theory, and complex multiplication. Wiley (1997)
21. De Feo, L.: Mathematics of isogeny based cryptography (2017), http://arxiv.org/abs/1711.04062
22. De Feo, L., Galbraith, S.D.: SeaSign: Compact isogeny signatures from class group actions. to appear in EuroCrypt 2019 (2019), https://eprint.iacr.org/2018/824
23. De Feo, L., Jao, D., Plût, J.: Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. Journal of Mathematical Cryptology **8**(3), 209–247 (2014)
24. Delfs, C., Galbraith, S.D.: Computing isogenies between supersingular elliptic curves over $\mathbb{F}_p$. Designs, Codes and Cryptography **78**(2), 425–440 (Feb 2016). https://doi.org/10.1007/s10623-014-0010-1
25. Doliskani, J., Pereira, G.C.C.F., Barreto, P.S.L.M.: Faster cryptographic hash function from supersingular isogeny graphs. Cryptology ePrint Archive, Report 2017/1202 (2017), https://eprint.iacr.org/2017/1202
26. Dwork, C., Naor, M.: Pricing via processing or combatting junk mail. In: CRYPTO. pp. 139–147 (1992)

27. Eisenträger, K., Hallgren, S., Lauter, K., Morrison, T., Petit, C.: Supersingular isogeny graphs and endomorphism rings: Reductions and solutions. In: Nielsen, J.B., Rijmen, V. (eds.) Advances in Cryptology – EUROCRYPT 2018. pp. 329–368. Springer International Publishing (2018)
28. Freeman, D., Scott, M., Teske, E.: A taxonomy of pairing-friendly elliptic curves. Journal of Cryptology **23**(2), 224–280 (2010). https://doi.org/10.1007/s00145-009-9048-z
29. Galbraith, S.D.: Mathematics of Public Key Cryptography. Cambridge University Press (2012)
30. Galbraith, S.D., Hess, F., Smart, N.P.: Extending the GHS Weil descent attack. In: Advances in cryptology–EUROCRYPT 2002 (Amsterdam), Lecture Notes in Computer Science, vol. 2332, pp. 29–44. Springer, Berlin (2002)
31. Galbraith, S.D., Petit, C., Shani, B., Ti, Y.B.: On the security of supersingular isogeny cryptosystems. In: Advances in Cryptology–ASIACRYPT 2016: 22nd International Conference on the Theory and Application of Cryptology and Information Security. pp. 63–91. Springer (2016)
32. Guralnick, R.M., Mller, P.: Exceptional polynomials of affine type. Journal of Algebra **194**(2), 429–454 (1997). https://doi.org/10.1006/jabr.1997.7028
33. Jao, D., De Feo, L.: Towards Quantum-Resistant cryptosystems from supersingular elliptic curve isogenies. In: Yang, B.Y. (ed.) Post-Quantum Cryptography. Lecture Notes in Computer Science, vol. 7071, pp. 19–34. Springer Berlin / Heidelberg, Berlin, Heidelberg (2011). https://doi.org/10.1007/978-3-642-25405-5_2
34. Jao, D., LeGrow, J., Leonardi, C., Ruiz-Lopez, L.: A polynomial quantum space attack on CRS and CSIDH. In: MathCrypt 2018 (2018), to appear
35. Jao, D., Soukharev, V.: A subexponential algorithm for evaluating large degree isogenies. In: ANTS IX: Proceedings of the Algorithmic Number Theory 9th International Symposium. Lecture Notes in Computer Science, vol. 6197, pp. 219–233. Springer, Berlin, Heidelberg (2010). https://doi.org/10.1007/978-3-642-14518-6_19
36. Kirschmer, M., Voight, J.: Algorithmic enumeration of ideal classes for quaternion orders. SIAM Journal on Computing **39**(5), 1714–1747 (2010). https://doi.org/10.1137/080734467
37. Kitaev, A.Y.: Quantum measurements and the abelian stabilizer problem. arXiv preprint quant-ph/9511026 (1995), https://arxiv.org/abs/quant-ph/9511026
38. Kohel, D.: Endomorphism rings of elliptic curves over finite fields. Ph.D. thesis, University of California at Berkley (1996)
39. Kohel, D.R., Lauter, K., Petit, C., Tignol, J.P.: On the quaternion-isogeny path problem. LMS Journal of Computation and Mathematics **17**(A), 418–432 (2014)
40. Koshiba, T., Takashima, K.: Pairing cryptography meets isogeny: A new framework of isogenous pairing groups. Cryptology ePrint Archive, Report 2016/1138 (2016), https://eprint.iacr.org/2016/1138
41. Kuperberg, G.: A subexponential-time quantum algorithm for the dihedral hidden subgroup problem. SIAM Journal of Computing **35**(1), 170–188 (2005)
42. Kuperberg, G.: Another Subexponential-time Quantum Algorithm for the Dihedral Hidden Subgroup Problem. In: Severini, S., Brandao, F. (eds.) 8th Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC 2013). Leibniz International Proceedings in Informatics (LIPIcs), vol. 22, pp. 20–34. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany (2013). https://doi.org/10.4230/LIPIcs.TQC.2013.20
43. Lenstra, A.K., Wesolowski, B.: A random zoo: sloth, unicorn, and trx. IACR Cryptology ePrint Archive **2015**, 366 (2015). https://doi.org/cr.org/2015/366

44. Mahmoody, M., Moran, T., Vadhan, S.: Publicly verifiable proofs of sequential work. In: Proceedings of the 4th conference on Innovations in Theoretical Computer Science. pp. 373–388. ACM (2013), http://www.cs.cornell.edu/~mohammad/files/papers/15%20TimeStamp.pdf
45. Menezes, A., Vanstone, S., Okamoto, T.: Reducing elliptic curve logarithms to logarithms in a finite field. In: STOC '91: Proceedings of the twenty-third annual ACM symposium on Theory of computing. pp. 80–89. ACM, New York, NY, USA (1991). https://doi.org/10.1145/103418.103434
46. Mestre, J.F.: La méthode des graphes. Exemples et applications. In: Proceedings of the international conference on class numbers and fundamental units of algebraic number fields (Katata, 1986). Nagoya University, Nagoya (1986), http://boxen.math.washington.edu/msri06/refs/mestre-method-of-graphs/mestre-fr.pdf
47. Network, C.: Chia network. https://chia.net/
48. Petit, C., Lauter, K.: Hard and easy problems for supersingular isogeny graphs. Cryptology ePrint Archive, Report 2017/962 (2017), http://eprint.iacr.org/2017/962
49. Pierrot, C., Wesolowski, B.: Malleability of the blockchain's entropy. Cryptography and Communications **10**(1), 211–233 (Jan 2018). https://doi.org/10.1007/s12095-017-0264-3
50. Pietrzak, K.: Simple Verifiable Delay Functions. In: Blum, A. (ed.) 10th Innovations in Theoretical Computer Science Conference (ITCS 2019). Leibniz International Proceedings in Informatics (LIPIcs), vol. 124, pp. 60:1–60:15. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany (2018). https://doi.org/10.4230/LIPIcs.ITCS.2019.60
51. Pizer, A.K.: Ramanujan graphs and Hecke operators. Bulletin of the American Mathematical Society (N.S.) **23**(1) (1990). https://doi.org/10.1090/S0273-0979-1990-15918-X
52. Pizer, A.K.: Ramanujan graphs. In: Computational perspectives on number theory (Chicago, IL, 1995), AMS/IP Stud. Adv. Math., vol. 7. Amer. Math. Soc., Providence, RI (1998)
53. Rabin, M.O.: Transaction protection by beacons. Journal of Computer and System Sciences, 27(2):256 ?267 (1983)
54. Regev, O.: A subexponential time algorithm for the dihedral hidden subgroup problem with polynomial space. arXiv:quant-ph/0406151 (Jun 2004), http://arxiv.org/abs/quant-ph/0406151
55. Rivest, R.L., Shamir, A., Wagner, D.A.: Time-lock puzzles and timed-release crypto (1996)
56. Silverman, J.H.: The arithmetic of elliptic curves, Graduate Texts in Mathematics, vol. 106. Springer-Verlag, New York (1992)
57. Sutherland, A.: Elliptic curves. Lecture notes from a course (18.783) at MIT (2017), http://math.mit.edu/classes/18.783/2017/lectures
58. The Sage Developers: SageMath, the Sage Mathematics Software System (Version 8.0) (2018), https://www.sagemath.org
59. Vélu, J.: Isogénies entre courbes elliptiques. Comptes Rendus de l'Académie des Sciences de Paris **273**, 238–241 (1971)
60. Vignéras, M.F.: Arithmétique des Algèbres de Quaternions, vol. 800. Springer Berlin Heidelberg (1980). https://doi.org/10.1007/bfb0091027
61. Voight, J.: Quaternion algebras (2018), https://math.dartmouth.edu/~jvoight/quat-book.pdf
62. Washington, L.C.: Elliptic curves: Number theory and cryptography, 2nd ed. CRC Press (2008)

63. Waterhouse, W.C.: Abelian varieties over finite fields. Annales Scientifiques de l'École Normale Supérieure **2**(4), 521–560 (1969)
64. Wesolowski, B.: Efficient verifiable delay functions. Cryptology ePrint Archive, Report 2018/623 (2018), https://eprint.iacr.org/2018/623
65. Yoo, Y., Azarderakhsh, R., Jalali, A., Jao, D., Soukharev, V.: A post-quantum digital signature scheme based on supersingular isogenies. In: Kiayias, A. (ed.) Financial Cryptography and Data Security. pp. 163–181. Springer International Publishing, Cham (2017)