Towards a Quantum-resistant Weak Verifiable Delay Function

Thomas Decru[1], Luciano Maino[2], and Antonio Sanso[3]

[1] COSIC, KU Leuven
[2] University of Bristol
[3] Ethereum Foundation

Abstract. In this paper, we present a new quantum-resistant weak Verifiable Delay Function based on a purely algebraic construction. Its delay depends on computing a large-degree isogeny between elliptic curves, whereas its verification relies on the computation of isogenies between products of two elliptic curves. One of its major advantages is its expected fast verification time. However, it is important to note that the practical implementation of our theoretical framework poses significant challenges. We examine the strengths and weaknesses of our construction, analyze its security and provide a proof-of-concept implementation.

Keywords: Verifiable Delay Function · Post-Quantum · Isogeny · Abelian Surface · Elliptic Curve Product.

1 Introduction

A Verifiable Delay Function (VDF) is a cryptographic primitive designed to take a prescribed amount of time t to compute, regardless of the parallel computing power available, while still being easy to verify once the computation is complete. VDFs are used in various applications, such as random number generation and blockchain consensus algorithms, where a delay is needed to ensure that certain operations cannot be performed too quickly. The seminal paper on VDFs, "Verifiable Delay Functions", was published in 2018 by Boneh, Bonneau, Bünz and Fisch [9]. In the paper, the authors introduce the concept of a VDF and describe its potential uses in various applications including auction protocols, proof-of-work systems, and secure multiparty computation. The first efficient VDFs were the ones proposed by Pietrzak [42] and Wesolowski [50]; both VDFs are based on exponentiation in a group of unknown order. We refer to [10] for a survey about these VDFs. Driven by the open problem of finding a VDF that is also quantum resistant, De Feo, Masson, Petit and Sanso [25] employed chains of supersingular isogenies as "sequential slow" functions in order to build their VDF. However, given the usage of bilinear pairing, this isogeny-based VDF is not quantum resistant but only provides some *quantum annoyance*. Proving knowledge of isogenies

Author list in alphabetical order; see https://www.ams.org//profession/leaders/CultureStatement04.pdf.

has a rich history of research (see for instance [20,7]), but none of the techniques seem to allow for a natural instantiation of a VDF.

Boneh, Bonneau, Bünz and Fisch [9], and independently Döttling, Garg, Malavolta, and Vasudevan [28] proposed the usage of SNARGs for constructing a VDF. In [17], Chavez-Saab, Rodríguez-Henríquez and Tibouchi describe an isogeny-based VDF that is quantum resistant based on the SNARG approach. Also, in [47], Tan, Sharma, Li, Szalachowski and Zhou report a VDF built over a sequential variant of the zero-knowledge proof system ZKBoo [33].

Our contribution. In this paper, we present a quantum-resistant *weak* VDF, which is a VDF where a certain amount of parallelism is needed to give an advantage to the evaluator [9, Definition 5]. Our construction is based upon both isogenies between supersingular elliptic curves and Kani's criterion [34]. Kani's criterion determines whether isogenies originating from elliptic products have split codomain. In our case, this criterion is leveraged in a constructive manner, in contrast to previous attacks [14,37,43] against the Supersingular Isogeny Diffie-Hellman key exchange protocol (SIDH) [24] and its instantiation SIKE [2]. While there have been other attempts to build quantum-resistant VDFs [17,47], to the best of our knowledge, this is the first instance where a quantum-resistant VDF has been constructed without relying on SNARG.

Our VDF is inherently noninteractive and does not have the limitation present in [25], where the time required for setting up public parameters is similar to the time required for evaluating the function.

However, our VDF faces two challenges: its *weakness* and the need of curves with unknown endomorphism ring as input. In our case, being weak means that Eval will require $\mathcal{O}(t)$ parallelism to run in parallel time t.

Sampling random supersingular elliptic curves over finite fields of cryptographic size without giving information about the endomorphism ring is necessary to ensure the security of the elliptic curve used in the Eval operation. Currently, finding a way to do this without relying on a trusted authority is an open problem in supersingular isogeny-based cryptography [11,40]. In [4], Basso, Codogni, Connolly, De Feo, Fouotsa, Lido, Morrison, Panny, Patranabis and Wesolowski suggest methods for creating such curves defined over a finite field $\mathbb{F}_{p^2}$ through a trusted setup. Nevertheless, engaging in a trusted setup for every single input is not a practical solution for us. Trusted setups often involve complex procedures and require the involvement of multiple parties or authorities, making them cumbersome to execute on a regular basis. In summary, the weakness of the VDF is a drawback, while the requirement for curves with unknown endomorphism rings as input is a significant obstacle.

Technical preview. Let $E_0/\mathbb{F}_p$ be a supersingular elliptic curve and ℓ an odd prime, such that there are two horizontal ℓ-isogenies $\psi : E_0 \to E_1$ and $\psi' : E_0 \to E_1'$. If the ℓ-torsion of E_0 is only defined over $\mathbb{F}_{p^{\ell-1}}$, then computing these isogenies is expensive, even with parallelization, and they will determine the delay of our weak VDF. On the other hand, one can rapidly verify this computation in dimension two by asking for the evaluation of ψ and ψ' on

$E_0[N]$ for certain smooth $N > \ell$ (e.g. N is some power of two). By choosing ℓ and N appropriately, the gap between evaluation and verification is exponential.

Outline. This paper is organized as follows. In Section 2, we give a mathematical foundation for understanding the concepts employed in the manuscript, as well as the definition of a weak VDF. Section 3, the main focus of the paper, provides a detailed description of our weak VDF. Section 4 to Section 6 present thorough analysis of correctness, soundness and sequentiality. Finally, we draw conclusions in Section 7.

Notation. We will call a prime ℓ a *safe* prime if $k = \frac{\ell-1}{2}$ is also an (odd) prime. The prime k is then necessarily a Sophie-Germain prime. The Legendre symbol $\left(\frac{a}{b}\right)$ is used to denote whether a is a quadratic residue modulo b or not. Two prime-field elements $a, b \in \mathbb{F}_p$ will be compared as $a <_{\mathbb{Z}} b$ if their canonical lifts $\overline{a}, \overline{b} \in \mathbb{Z} \cap [0, p-1]$ satisfy $\overline{a} < \overline{b}$, and analogously for $>_{\mathbb{Z}}$. For a point P on an elliptic curve E, we will denote its x-coordinate (respectively y-coordinate) by $x(P)$ (respectively $y(P)$). We will use the term "taking t time to compute" when referring to the evaluation of a polynomial-sized arithmetic circuit with a maximum depth of t, specifying the breadth of the circuit when needed.

Acknowledgments. We would like to thank the CRYPTO 2023 and LATINCRYPT 2023 anonymous reviewers; in particular the CRYPTO reviewer pointing out the attack described in Subsection 3.2. We would also like to thank Wouter Castryck, Chloe Martindale and Frederik Vercauteren for fruitful discussions and feedback on a previous version of this manuscript. This work was supported in part by CyberSecurity Research Flanders with reference number VR20192203, by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement ISOCRYPT - No. 101020788) and by the UK Engineering and Physical Sciences Research Council (EPSRC) Centre for Doctoral Training (CDT) in Trust, Identity, Privacy and Security in Large-scale Infrastructures (TIPS-at-Scale) at the Universities of Bristol and Bath.

2 Preliminaries

In this section, we will discuss some properties related to isogenies and weak VDFs. In general, we will assume the characteristic of the field we work over to be a prime $p > 3$, although certain results generalize beyond this restriction.

2.1 Elliptic Curves and Their Representation

Elliptic curves are smooth projective algebraic curves of genus one with a fixed given point $\mathcal{O}$. Any such curve can be written in long Weierstraß form and then $\mathcal{O}$ is the (only) point at infinity. Often, the curve is given as an affine equation without explicit mention of $\mathcal{O}$; e.g. the Montgomery form of an elliptic curve E_A is given by

$$E_A/K : y^2 = x^3 + Ax^2 + x,$$

where K is the field we work over and A is an element of this field. An elliptic curve comes equipped with a natural group law and the point at infinity $\mathcal{O}_E$ is the neutral element of this group. The K-rational points of E (which include $\mathcal{O}_E$) are denoted by $E(K)$.

In isogeny-based cryptographic settings, elliptic curves are typically only considered up to isomorphism. Two elliptic curves are isomorphic over $\overline{K}$ if and only if they have the same j-invariant $j \in K$. The j-invariant of an elliptic curve E in Montgomery form is denoted by $j(E_A)$ and given by

$$j(E_A) = \frac{(A^2-3)^3}{A^2-4}.$$

Given a j-invariant $j \neq 1728$, we will define the Weierstraß form

$$E(j) : y^2 + xy = x^3 - \frac{36}{j-1728}x - \frac{1}{j-1728}$$

as *the canonical representation* of E in the isomorphism class of E. The canonical representation isomorphism $\iota_j : E \to E(j)$ is easy to compute for any elliptic curve E. An elliptic curve is in canonical form if $E = E(j(E))$. Other forms of elliptic curves than $E(j)$ are often preferred for computational purposes. For instance, the Montgomery form E_A allows efficient x-only arithmetic in the group by means of the Montgomery ladder [6]. From the expression $j(E_A)$ above though, it is clear that for any given j-invariant there may be up to six distinct Montgomery coefficients A. Additionally, one cannot represent every j-invariant as an elliptic curve in Montgomery form without using field extensions, hence the Montgomery coefficient is less useful from a representational point of view. For more information about elliptic curves in general, the book by Silverman is a staple reference [44].

2.2 Isogenies

An isogeny $\phi : E \to E'$ between elliptic curves is a surjective morphism with finite kernel. In this paper, we will restrict ourselves mostly to separable isogenies. Assuming kernel points are considered over the algebraic closure, it holds that $\deg \phi = \# \ker \phi$ for all separable isogenies. An example of an isogeny is the multiplication-by-n map, given by $[n] : E \to E$, $P \mapsto [n]P$. This isogeny is of degree n^2 and its kernel is denoted by $E[n]$. An endomorphism is a homomorphism from an elliptic curve to itself. The endomorphism ring $\mathrm{End}_K(E)$ of an elliptic curve is the ring of all endomorphisms of E defined over the field K.

There are two options for the group structure of $E[p]$, namely $E[p] \cong \{0\}$ or $E[p] \cong \mathbb{Z}/p\mathbb{Z}$. In the former case, the elliptic curves are called supersingular, whereas in the latter case, they are called ordinary. We will restrict ourselves to supersingular elliptic curves and isogenies between them, since it is significantly easier to generate supersingular elliptic curves with certain given orders. For instance, a supersingular elliptic curve has order $p+1$ over $\mathbb{F}_p$.

For a supersingular elliptic curve $E/\mathbb{F}_p$, either $\mathrm{End}_{\mathbb{F}_p}(E)$ equals $\mathbb{Z}[\sqrt{-p}]$ or $\mathbb{Z}[(1+\sqrt{-p})/2]$. In the former case, the elliptic curve E is said to be *on the floor*, whereas in the latter case, the elliptic curve is said to be *on the surface*. An isogeny $\phi : E \to E'$ is said to be *horizontal* in this context if $\mathrm{End}_{\mathbb{F}_p}(E) = \mathrm{End}_{\mathbb{F}_p}(E')$ (i.e. E and E' need to either be both on the floor, or both on the surface). We will make use of the following theorem, where two isogenies are considered distinct if they have different kernel.

Theorem 1. *Let $p > 3$ be a prime such that $p \equiv 3 \bmod 4$, and ℓ an odd prime such that $\left(\frac{-p}{\ell}\right) = 1$. If $E/\mathbb{F}_p$ is a supersingular elliptic curve, then there are exactly two distinct $\mathbb{F}_p$-rational horizontal isogenies of degree ℓ with E as domain.*

Proof. This is part of [26, Theorem 2.7]. □

In CSIDH [16], they choose p such that $\#E(\mathbb{F}_p) = p+1$ has many small odd prime factors ℓ_i. For each ℓ_i, the two horizontal ℓ_i-isogenies are then not only $\mathbb{F}_p$-rational, but they are cyclic with kernel generators in $\mathbb{F}_p$ and $\mathbb{F}_{p^2}$. With a good choice of representation, both isogenies can be computed from the x-coordinate of their respective kernel generator using arithmetic over $\mathbb{F}_p$ only. Generically, however, these two $\mathbb{F}_p$-rational horizontal isogenies have kernel generators in $E(\mathbb{F}_{p^e})$, for some $e \leq \ell - 1$.

In our protocol, we will post-compose $\mathbb{F}_p$-rational horizontal isogenies with an isomorphism onto the canonical form of the image curve. Technically, the resulting isogenies are not $\mathbb{F}_p$-rational anymore because of this isomorphism. However, since we work with curves of unknown endomorphism ring, we can discard the case where the j-invariant of the starting curve is either 0 or 1728. As a result, the two horizontal isogenies are still distinct [1, Lemma 3.11]. For more general background regarding isogenies in a cryptographic setting, we refer the reader to the notes by De Feo [23].

2.3 Isogenies between Abelian Surfaces

Abelian surfaces are abelian varieties of dimension two, which can be seen as a generalization of (necessarily one-dimensional) elliptic curves. In the context of isogeny-based cryptography, it is necessary to equip them with a principal polarization (abbreviated as p.p. from now on). We will not elaborate on the notion of polarizations, but refer the interested reader to [45, Section 2.2] for more details.

All p.p. abelian surfaces (up to $\overline{K}$-isomorphism) are either products of two elliptic curves or Jacobians of genus-2 curves. Arithmetic on a product of elliptic curves (E_1, E_2) is simply arithmetic on the two curves componentwise; e.g. for $(P_1, P_2) \in (E_1, E_2)$ we can compute the multiplication-by-n map as $([n]P_1, [n]P_2) \in (E_1, E_2)$. A genus-2 curve C is a smooth projective algebraic curve of genus two. Over a field of positive odd characteristic p, such a curve can be given by an affine equation of the form $C : y^2 = F(x)$, where $F(x)$ is a degree-six polynomial, together with two points at infinity (which may only exist over a quadratic field extension). From the points on this curve, one can also construct a group called the Jacobian of the genus-2 curve. Remark that to construct all K-rational elements of this Jacobian, one needs to consider all K'-rational points on C for a quadratic extension $K' \supseteq K$. For an explicit construction of this group law, see for example [19].

Just as in the case of elliptic curves, isogenies between p.p. abelian surfaces are surjective morphisms with finite kernel. In order to ensure that the isogeny is compatible with the chosen polarizations of the domain and codomain, this finite kernel will have to satisfy certain conditions. A sufficient condition is that the kernel of the isogeny has to be maximal isotropic with regards to the Weil pairing. For instance, if $\Psi : A \to A'$ is an isogeny between p.p. abelian surfaces with kernel isomorphic to $\mathbb{Z}/3 \oplus \mathbb{Z}/3$, then for any two elements D_1, D_2 in $\ker \Psi$ it must hold that $e_3(D_1, D_2) = 1$. A group satisfying these conditions is called a $(3,3)$-*subgroup* and the associated isogeny a $(3,3)$-*isogeny*.

A theorem by Kani proves under which specific conditions an isogeny Φ with domain $E_1 \times E_2$ has a codomain which is again a product of elliptic curves. These conditions connect E_1 and E_2 by means of another (one-dimensional) isogeny. This criterion underlies Theorem 2 formulated in Section 4, which we use to prove correctness of our protocol. If $F_1 \times F_2$ is the codomain of Φ, then we say that Φ has *product codomain passing through* F_i. For an introductory framework with regards to higher-dimensional isogenies in a cryptographic setting, see for example [15].

2.4 Weak VDFs

For the sake of being self-contained, we briefly recall the notion of *weak VDF* introduced by Boneh, Bonneau, Bünz and Fisch [9]. The main difference between a VDF and a weak VDF lies in the parallelization capabilities given to evaluators: in a weak VDF, an evaluator needs arithmetic circuits of breadth $\mathcal{O}(\mathbf{poly}(t))$ to achieve the best strategy, where t indicates the delay expected.

Definition 1. *A weak VDF $V = (\mathsf{Setup}, \mathsf{Eval}, \mathsf{Verify})$ consists of a triple of algorithms as follows:*

- $(\mathsf{ek}, \mathsf{vk}) \leftarrow \mathsf{Setup}(\lambda, t)$: *is a randomized algorithm that takes a security parameter λ and a delay parameter t as input, and outputs an evaluation key ek and a verification key vk. The input (λ, t) also defines a domain $\mathcal{X}$ and a codomain $\mathcal{Z}$. Also, Setup should run in $\mathcal{O}(\boldsymbol{poly}(\lambda))$.*
- $(z \in \mathcal{Z}, \pi) \leftarrow \mathsf{Eval}(\mathsf{ek}, x \leftarrow \mathcal{X})$: *on input the evaluation key ek and $x \in \mathcal{X}$, returns $z \in \mathcal{Z}$ and a proof π. This algorithm must run in time t on an arithmetic circuit of breadth $\mathcal{O}(\boldsymbol{poly}(t, \lambda))$.*
- $\{\mathsf{True}, \mathsf{False}\} \leftarrow \mathsf{Verify}(\mathsf{vk}, x, z, \pi)$: *checks whether the output z corresponds to the input x. This algorithm must run in $\mathcal{O}(\boldsymbol{poly}(\log t, \lambda))$ time.*

Furthermore, V must satisfy the following properties:

- ***Correctness:*** *A weak VDF is correct if, for all parameters λ, t, an honest evaluation of* Eval *always passes the check made by* Verify*.*
- ***Soundness:*** *A weak VDF is sound if the probability of marking a wrong evaluation as correct is negligible in the security parameter λ.*
- ***Sequentiality:*** *To define sequentiality, we need to introduce the following game applied to the adversary $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$:*

$$\begin{aligned}(\mathsf{ek}, \mathsf{vk}) &\leftarrow \mathsf{Setup}(\lambda, t)\\ L &\leftarrow \mathcal{A}_0(\mathsf{ek}, \mathsf{vk})\\ x &\leftarrow_{\$} \mathcal{X}\\ z_{\mathcal{A}} &\leftarrow_{\$} \mathcal{A}_1(L, \mathsf{ek}, \mathsf{vk}, x)\end{aligned}$$

The adversary $\mathcal{A}$ wins the game if $z_{\mathcal{A}} = z$, where $(z, \pi) = \mathsf{Eval}(\mathsf{ek}, x)$. Given $\sigma(t)$ and $p(t)$, the weak VDF V is $(p, \sigma)-$sequential if no pair of randomized algorithms $\mathcal{A}_0$, which runs in time $\mathcal{O}(\boldsymbol{poly}(\lambda, t))$, and $\mathcal{A}_1$, which runs in time strictly less than $\sigma(t)$ on an arithmetic circuit of breadth $p(t)$, can win the security game above with probability greater than $\mathsf{negl}(\lambda)$.

3 The VDF

In this section, we give a high-level description of our weak VDF. Once the evaluation key ek and verification key vk have been sampled, the input space consists of $\mathcal{E}\ell\ell_p$, the set of all j-invariants corresponding to supersingular elliptic curves over $\mathbb{F}_p$ whose $\mathbb{F}_{p^2}$-endomorphism ring is unknown. Currently, finding a way to sample such curves at random is an open problem in supersingular isogeny-based cryptography [11,40]. We define $\mathsf{Gen2b}$ to be a deterministic algorithm that, on input a supersingular elliptic curve and a positive integer b, outputs a basis of the 2^b-torsion.

We recall that λ is a security parameter, t is a delay parameter, z is the output and π is the proof of the output.

```
$(\mathsf{ek}, \mathsf{vk}) \leftarrow \mathsf{Setup}(\lambda, t)$:
 1. Sample a random safe prime $\ell \sim t$ and define $k = (\ell - 1)/2$.
 2. Let $b > \lambda$ such that $2^b = c^2\ell + d^2$ for some coprime positive integers $c, d \in \mathbb{N}$.
 3. Construct a random $\lambda \log^3(t)/2$-bit prime $p$ such that
    (a) $p \equiv -1 \bmod 2^b cd$;
    (b) $p \equiv 1 \bmod k$ and $2^{\frac{p-1}{k}} \not\equiv 1 \bmod p$.
    (c) the order of $-p$ in $\mathbb{F}_\ell^*$ equals $k$;
    (d) $\left(\frac{-p}{\ell}\right) = 1$;
 4. $\mathsf{ek} = (p, b, \ell)$, $\mathsf{vk} = (p, b, \ell, c, d)$.
```

```
$(z, \pi) \leftarrow \mathsf{Eval}(j \leftarrow_{\$} \mathcal{E}\ell\ell_p, \mathsf{ek})$:
 1. $E_0 \leftarrow E(j)$, $P_0, Q_0 \leftarrow \mathsf{Gen2b}(E_0, b)$.
```

2. Compute the two (distinct) horizontal ℓ-isogenies $\psi : E_0 \to E_1/\mathbb{F}_p$ and $\psi' : E_0 \to E_1'/\mathbb{F}_p$, where E_1 and E_1' are in canonical form, as well as $P_1 = \psi(P_0)$, $Q_1 = \psi(Q_0)$, $P_1' = \psi'(P_0)$ and $Q_1' = \psi'(Q_0)$.
3. If $j(E_1) >_{\mathbb{Z}} j(E_1')$, then swap $(E_1, P_1, Q_1) \leftrightarrow (E_1', P_1', Q_1')$.
4. $z \leftarrow x(P_1) \mid\mid x(Q_1) \mid\mid x(P_1') \mid\mid x(Q_1')$.
5. $\pi \leftarrow j(E_1) \mid\mid j(E_1') \mid\mid y(P_1) \mid\mid y(Q_1) \mid\mid y(P_1') \mid\mid y(Q_1')$.

$\{\mathsf{True}, \mathsf{False}\} \leftarrow \mathsf{Verify}(j, z, \pi, \mathsf{vk})$:

1. $E_0 \leftarrow E(j)$, $P_0, Q_0 \leftarrow \mathsf{Gen2b}(E_0, b)$.
2. Verify that $j(E_1) <_{\mathbb{Z}} j(E_1')$.[4]
3. $E_1 \leftarrow E(j(E_1))$, $E_1' \leftarrow E(j(E_1'))$.
4. Verify that $P_1, Q_1 \in E_1(\mathbb{F}_{p^2})$, and $P_1', Q_1' \in E_1'(\mathbb{F}_{p^2})$.
5. Verify that the subgroups $\langle([d]P_1, [c\ell]P_0), ([d]Q_1, [c\ell]Q_0)\rangle \subset E_1 \times E_0$ and $\langle([d]P_1', [c\ell]P_0), ([d]Q_1', [c\ell]Q_0)\rangle \subset E_1' \times E_0$ define two kernels of $(2^b, 2^b)$-isogenies Φ and Φ', respectively, having product codomain passing through E_0.
5. Verify that, for all $S \in E_1[c]$, the projections of $\Phi(S, 0)$ and $\Phi'(S, 0)$ onto E_0 are equal to the identity.
6. Verify that, for all $S \in E_0[d]$, the projections of $\Phi(0, S)$ and $\Phi'(0, S)$ onto E_0 are equal to the identity.

For now, we will assume that the evaluation of an ℓ-isogeny in this setting is expensive, even with access to a large amount of parallel processors. We will elaborate on this in Section 6 when discussing sequentiality but will explain the choices in the protocol first.

Remark 1. Remark that $\mathsf{ek}, \mathsf{vk}, z, \pi$ can be noticeably compressed in bitsize; e.g. the y-coordinates of P_1, Q_1, P_1' and Q_1' can be compressed to four bits in the classical way. [5] For the clarity of exposition, we elect to omit these details involving bandwidth requirements.

3.1 The Conditions in Setup

The condition $\left(\frac{-p}{\ell}\right) = 1$ ensures that there exist two horizontal ℓ-isogenies, see Theorem 1. The condition $-p$ having order k in $\mathbb{F}_\ell^*$ implies that the minimal field extension over which an ℓ-torsion point is defined is $\mathbb{F}_{p^k}$. Indeed, if E_0 is a supersingular elliptic curve defined over $\mathbb{F}_p$, then $\#E_0(\mathbb{F}_{p^k}) = p^k + 1$. Since $-p$ has order k in $\mathbb{F}_\ell^*$, we have that $\ell \mid (p^k + 1)$. The field $\mathbb{F}_{p^k}$ is the minimal field extension since it is an extension of prime degree of $\mathbb{F}_p$ and $\ell \nmid (p + 1)$. Finally, the form of p implies that all 2^b-, c- and d-torsion is $\mathbb{F}_{p^2}$-rational, which will allow fast verification.

[4] Checking if $j(E_1)$ is smaller than $j(E_1')$ implicitly verifies that $j(E_1), j(E_1') \in \mathbb{F}_p$.

[5] Given that they serve as part of kernel generators for verifying a two-dimensional isogeny, they can actually be compressed to a combined two bits.

The conditions $k \mid p-1$ and $2^{\frac{p-1}{k}} \not\equiv 1 \bmod p$ are needed to ensure that the polynomial x^k+2 is irreducible over $\mathbb{F}_p[x]$. Since $2^{\frac{p-1}{k}} \not\equiv 1 \bmod p$, 2 does not admit a k-th root over $\mathbb{F}_p$, which in turn proves that x^k+2 is irreducible over $\mathbb{F}_p[x]$. The polynomial x^k+2 is then used to define the field $\mathbb{F}_{p^k}$, i.e. $\mathbb{F}_{p^k} = \mathbb{F}_p[x]/(x^k+2)$. This condition is technically not needed but ensures that we do not need to waste time searching for irreducible polynomials to define $\mathbb{F}_{p^k}$.

3.2 The Size of p

The computation of the horizontal ℓ-isogenies correspond to the action of $\mathfrak{l} = (\ell, \pi^k - 1)$ and $\bar{\mathfrak{l}} = (\ell, \pi^k + 1)$ in the class group $\mathrm{Cl}(\mathcal{O})$ of the $\mathbb{F}_p$-endomorphism ring $\mathcal{O}$ of E_0. We will focus on $\mathfrak{l}$, the other case is completely analogously. Assuming access to a sufficiently large quantum computer, the relation lattice for a given set of generators - say $\mathfrak{l}, \mathfrak{l}_1, \ldots, \mathfrak{l}_{d-1}$ - of $\mathrm{Cl}(\mathcal{O})$ can be computed. This means that an adversary could try to simplify the computation of the ℓ-isogeny by means of finding an equivalent element $\mathfrak{l} = \mathfrak{l}_1^{e_1} \cdot \ldots \cdot \mathfrak{l}_{d-1}^{e_{d-1}}$ when seen as elements in $\mathrm{Cl}(\mathcal{O})$. Each $\mathfrak{l}_i$ in this product corresponds to a prime-degree isogeny, such that ideally the e_i are as small as possible. This is exactly how the CSI-FiSh signature scheme is made efficient [8].

To combat this, we can choose p to be large enough, such that any of the known lattice reduction algorithms takes time at least 2^λ to find a short vector of L^1-norm less than t. This implies that no reasonable lattice reduction can find an equivalent smooth-norm ideal corresponding to less than t sequential isogenies.

Following the argument of Panny [41], the standard lattice reduction algorithm which gives a trade-off between time spent reducing the lattice and the quality (read: norm) of the output vector is the BKZ algorithm. Assuming p is a μ-bit prime, our lattice has dimension d and covolume $2^{\mu/2}$, since the class group has order $\mathcal{O}(\sqrt{p})$. If we are looking for vectors bounded in L^1-norm by $t = 2^\tau$, we can deduce that the optimal trade-off happens for dimension $d \approx \mu/\tau$. The total runtime of the BKZ algorithm is then $2^{\mathcal{O}(2\mu/\tau^2)} \approx 2^{2\mu/\tau^2}$. Assuming BKZ is fully parallelizable, with access to arithmetic circuits of breadth $t = 2^\tau$, it runs in time $2^{2\mu/\tau^3}$. To ensure that this is still more than 2^λ, we must have that $2\mu/\tau^3 \geq \lambda$, or $\mu \geq \lambda \log(t)^3/2$.

Remark that this approach would lead a dishonest evaluator only to the codomain curve E_1, but this can be extended to also compute the images of P_0 and Q_0 as follows.

Write $R_0 = P_0 + Q_0$, such that $\langle R_0 \rangle$ is a cyclic group defining a descending 2^b-isogeny to a curve $E'/\mathbb{F}_{p^2}$. This curve is oriented by an order $\mathcal{O}'$ of conductor 2^b inside $\mathrm{End}(E_0)$; in particular its group action is compatible with the one at the surface. The class group relations can be obtained as well, and hence $\mathfrak{l}_1^{e_1} \cdot \ldots \cdot \mathfrak{l}_{d-1}^{e_{d-1}} \cap \mathcal{O}'$ can be rewritten as an equivalent ideal of smooth norm, say $\mathfrak{m}$. The image curve $\mathfrak{m}E'$ is then equivalent to $E_1/\langle R_1 \rangle$, with R_1 the image of R_0 under the isogeny ψ defined by $\mathfrak{l}$.

Furthermore, $E_1[2^b]$ contains two distinguished cyclic subgroups corresponding to the two eigenvalues of Frobenius. This means that on the level of subgroups, we can distinguish $\langle\psi(P_0)\rangle$ and $\langle\psi(Q_0)\rangle$ easily. Adding our cyclic subgroup $\langle\psi(R_0)\rangle$ as third piece of information, one can use the Weil-pairing and some linear algebra as in [31] to recover the exact images of P_0 and Q_0.

Remark 2. The aforementioned derivation of the size of p is extremely conservative. Not only does it assume full parallelizability of BKZ with no overhead, but it also assumes a dishonest evaluator can compute ℓ_i-isogenies in time $\mathcal{O}(1)$ for d distinct primes ℓ_i. In practice this will also come with a huge overhead, since our parameters are not set up such that both the ℓ_i and the field extension over which the ℓ_i-torsion is defined are simultaneously small.

3.3 Curves with Unknown Endomorphism Ring

If the endomorphism ring of the curve E given as input is known, there exists a polynomial-time algorithm that allows one to compute ℓ-isogenies without using the arithmetic on extension fields [37]: an attacker could extend $(\ell, \pi^k \pm 1)$ to a fractional ideal $I_\pm$ in the maximal order $\mathrm{End}(E_0)$. Then, computing an isogeny associated with $I_\pm$ has complexity $\mathcal{O}(\mathbf{poly}(\log p + C))$, where C is the bit-size of the representation of $\mathrm{End}(E_0)$ [36, Propositon 5].

To avoid this, it is needed to employ elliptic curves where the endomorphism ring remains unknown. Currently, one strategy is to depend on a “trusted party” to generate a random curve and then eliminate any sensitive information connected to it. Another option is to consider a distributed trusted-setup ceremony, as described in [4], which outlines a procedure for obtaining supersingular elliptic curves with an unknown endomorphism ring.

However, having a trusted setup for every single input is not a practical solution in this context. Indeed performing a trusted setup for each input would introduce significant overhead in terms of time, resources, and complexity. Additionally, frequent trusted setups can become prohibitively expensive, especially in scenarios where a large number of inputs need to be processed. Given these challenges, it becomes crucial to explore alternative methods that do not rely on a trusted setup as in [11,40].

3.4 The Role of the Security Parameter

The condition on b is needed to avoid that an attacker having access to the ℓ-modular polynomial $\Phi_\ell(X, Y)$ can break sequentiality with probability greater than $\mathsf{negl}(\lambda)$. The classical modular polynomial $\Phi_\ell(X, Y) \in \mathbb{Z}[X, Y]$ is a polynomial which vanishes on the j-invariants of every pair of elliptic curves which are ℓ-isogenous. The polynomial can be precomputed and stored in space $\mathcal{O}(\ell \log p)$ since it is a symmetric polynomial with bidegree $\ell + 1$. For any given $j(E)$, the univariate polynomial $\Phi_\ell(X, j(E))$ can be computed in parallel by using the Chinese remainder theorem, see for example [46].

The polynomial $\Phi_\ell(X, j(E))$ splits into linear factors over $\mathbb{F}_{p^2}$, and the $\mathbb{F}_p$-rational roots correspond to the $\mathbb{F}_p$-rational ℓ-isogenous curves. However, having the j-invariants of the two curves is not enough to pass Verify. Starting from

$E_0/\mathbb{F}_p$, once an ℓ-isogenous elliptic curve $E_1/\mathbb{F}_p$ has been computed via the evaluation of roots in classical modular polynomials, an attacker has to guess the image of the 2^b-torsion under the $\mathbb{F}_p$-rational isogeny. For instance, the attacker could proceed in the following way.

Let $\langle P_0, Q_0\rangle = E_0[2^b]$ be a basis of eigenvectors for the p-Frobenious endomorphism π. Since the eigenspaces are preserved by horizontal isogenies ψ, we know $\langle\psi(P_0)\rangle$ and $\langle\psi(Q_0)\rangle$. Since $e_{2^b}(\psi(P_0), \psi(Q_0)) = e_{2^b}(P_0, Q_0)^{\deg\psi}$, each guess of $\psi(P_0)$ corresponds to a unique guess of $\psi(Q_0)$. That is, given a $P_1 \in \langle\psi(P_0)\rangle$ and $Q_1 \in \langle\psi(Q_0)\rangle$, for each $s_p \in [0, 2^b - 1]$, an attacker will compute $s_Q \in [0, 2^b - 1]$ such that $e_{2^b}(P_1, Q_1)^{s_P s_Q} = e_{2^b}(P_0, Q_0)^{\deg\psi}$. Then, for each $([s_P]P_1, [s_Q]Q_1)$, he can check that it is the correct image of the (P_0, Q_0) under ψ running Verify. Since $b > \lambda$, the probability of guessing the right image is negligible in λ. We highlight that even if (P_0, Q_0) is not the basis provided as input, an attacker can perform computations with a basis of eigenvectors and then reconstruct the image of the provided basis via a discrete logarithm computation in $\mathbb{Z}/2^b\mathbb{Z}$, which is extremely efficient.

Remark 3. We stress that it is not clear exactly how well the parallelization of [46] performs in practice compared to the work we let our evaluator do. It may thus seem overly cautious to assume that an attacker has early access to $j(E_1)$. However, from pushing points through an isogeny, one can easily reconstruct the codomain curve as well (see for example [5]), which makes the evaluation of points a problem that is at least as hard as finding the codomain curve. Since the image points are needed to make use of Kani's criterion anyway, we thus see no argument to *not* put $j(E_1)$ as part of the proof, since other algorithms to compute it may be faster by a small constant factor. Additionally, Elkies algorithm to reconstruct the ℓ-isogeny from just $j(E_0)$ and $j(E_1)$ involves a recurrence relation of length $\mathcal{O}(\ell^2)$ (see [29]), which will be outperformed by our approach outlined in Section 6.

4 Correctness

The correctness of the scheme depends on the following result.

Theorem 2. *Let $\varphi_{N_1}: E_0 \to E_1$ and $\varphi_{N_2}: E_0 \to E_2$ be two isogenies of coprime degrees $\deg(\varphi_{N_1}) = N_1$ and $\deg(\varphi_{N_2}) = N_2$, and let $\langle P, Q\rangle$ be a basis of $E_0[N_1 + N_2]$. Then, the subgroup*

$$\langle([N_2]\varphi_{N_1}(P), [N_1]\varphi_{N_2}(P)), ([N_2]\varphi_{N_1}(Q), [N_1]\varphi_{N_2}(Q))\rangle \subset E_1 \times E_2,$$

is the kernel of an (N_1+N_2, N_1+N_2)-polarized isogeny Φ having product codomain endowed with the product polarization. Moreover, the isogeny Φ has matrix form

$$\begin{pmatrix} \widehat{\varphi_{N_1}} & -\widehat{\varphi_{N_2}} \\ f_{N_2} & \widehat{f_{N_1}} \end{pmatrix},$$

where the f_{N_i}'s are N_i-isogenies such that $\varphi_{N_2} \circ \widehat{\varphi_{N_1}} = f_{N_1} \circ f_{N_2}$.

Proof. This result is a consequence of Kani's criterion [34]. We refer to [37, Theorem 1] for a description of how the result is derived from [34]. □

In Verify, one has to check that the subgroups $\langle([d]P_1,[c\ell]P_0),([d]Q_1,[c\ell]Q_0)\rangle$ and $\langle([d]P_1',[c\ell]P_0),([d]Q_1',[c\ell]Q_0)\rangle$ define two kernels of $(2^b,2^b)$-isogenies having product codomains passing through E_0 and that the projections onto E_1 and E_1' contain the scalar multiplication $[c]$. Since the two checks are independent, let us focus uniquely on $\mathcal{K} \coloneqq \langle([d]P_1,[c\ell]P_0),([d]Q_1,[c\ell]Q_0)\rangle$.

Recall that $P_1=\psi(P_0)$, $Q_1=\psi(Q_0)$, where $\psi\colon E_0\to E_1$ is a horizontal ℓ-isogeny. Applying Theorem 2 with $\varphi_{N_1}=[c]\circ\psi$ and $\varphi_{N_2}=[d]$, we have that the $(2^b,2^b)$-isogeny Φ having kernel $\mathcal{K}$ has matrix form

$$\begin{pmatrix}[c]\circ\widehat{\psi} & -[d]\\ [d] & [c]\circ\psi\end{pmatrix}: E_1\times E_0\to E_0\times E_1.$$

The isogeny Φ clearly passes through E_0. Moreover, it is easy to check that, for all $S\in E_1[c]$, $\Phi(S,0)=(0,[d]S)$, which means that the projection onto E_1 contains the scalar multiplication $[c]$. Similarly, for $S\in E_0[d]$, $\Phi(0,S)=(0,[c]\psi(S))$. Evaluating a $(2^b,2^b)$-isogeny from a given kernel can be done in $\mathcal{O}(b\log b)$ $\mathbb{F}_p$-operations using the optimal strategies described in [2].

5 Soundness

In this section, we prove soundness assuming that $d^2>\ell$, where d and ℓ are as in the verification key vk in Section 3. In practice, the condition $d^2>\ell$ is trivially satisfied.

Theorem 3. *Let $d^2>\ell$. The weak VDF described in Section 3 is sound.*

Proof. Let ek, vk be the evaluation and verification keys, respectively, obtained via $\mathsf{Setup}(\lambda,t)$ on input some parameters λ and t. Given $j\in\mathcal{E}\ell\ell_p$, let (z,π) be any data such that $\mathsf{Verify}(j,z,\pi,\mathsf{vk})=\mathsf{True}$. We will prove that z has been honestly generated with overwhelming probability.

In what follows, we abide to notation used in Section 3. The first four lines in Verify ensure that any adversary cannot swap the points on E_1 and E_1' around and produce other valid outputs. Also, note that Verify performs two independent checks on the triples (E_1,P_1,Q_1) and (E_1',P_1',Q_1'). Hence, we will uniquely focus on the triple (E_1,P_1,Q_1); the other triple is analogous. Observe that the kernel $\langle([d]P_1,[c\ell]P_0),([d]Q_1,[c\ell]Q_0)\rangle$ defines a $(2^b,2^b)$-polarized isogeny Φ having product codomain $E_0\times F$ (up to polarized isomorphisms), for some supersingular elliptic curve F. In particular, we can write Φ in its matrix form

$$\begin{pmatrix}\alpha_{1,1} & \alpha_{1,2}\\ \alpha_{2,1} & \alpha_{2,2}\end{pmatrix}: E_1\times E_0\to E_0\times F,$$

where the $\alpha_{i,j}$ are isogenies making the matrix meaningful.

Since $\Phi\circ\hat{\tilde{\Phi}}=[2^b]$, $\deg(\alpha_{1,1})+\deg(\alpha_{1,2})=2^b$. Additionally, $\alpha_{1,1}=[c]\circ\mu_1$ and $\alpha_{1,2}=[d]\circ\mu_2$, which implies $c^2\deg(\mu_1)+d^2\deg(\mu_2)=2^b$. Since $c^2\ell+d^2=2^b$, we have $\deg(\mu_1) = \ell \pmod{d^2}$. As a consequence of $d^2 > \ell$, we have that $\deg(\mu_1) = \ell$ and $\deg(\mu_2) = 1$, that is $\alpha_{1,1} = [c] \circ \mu_1$ and $\alpha_{1,2} = [d]$ up to isomorphism.

In particular, the isogeny $\widehat{\mu_1}\colon E_0 \to E_1$ is an ℓ-isogeny between supersingular elliptic curves defined over $\mathbb{F}_p$. The codomains of nonhorizontal ℓ-isogenies are defined over $\mathbb{F}_{p^2} \setminus \mathbb{F}_p$ with overwhelming probability. To be more precise, the amount of supersingular elliptic curves over $\mathbb{F}_p$ is $\mathcal{O}(\sqrt{p})$, while the number of those over $\mathbb{F}_{p^2}$ is $\mathcal{O}(p)$. Therefore, the probability of E_1 being defined over $\mathbb{F}_p$ when $\widehat{\mu_1}$ does not correspond to a horizontal isogeny is $\mathcal{O}(1/\sqrt{p})$, which is negligible in λ. □

6 Sequentiality

The sequentiality of the weak VDF relies on the following assumption.

Assumption 1 *Let ℓ be a prime and p a $\lambda \log(\ell)^3/2$-bit prime, where λ is a security parameter. Let $E_0/\mathbb{F}_p$ be a supersingular elliptic curve with unknown endomorphism ring such that the minimal extension for an ℓ-torsion point of E_0 to be defined over is $\mathbb{F}_{p^k}$, where $k = (\ell-1)/2$ is a prime. Then, the best technique to evaluate a horizontal ℓ-isogeny with domain E_0 requires $\mathcal{O}(\ell \log \ell)$ $\mathbb{F}_p$-operations, even with access to a quantum computer and arithmetic circuits of breadth $\mathcal{O}(\textbf{poly}(\ell))$.*

Accurately defining wall-clock time in formal terms is a difficult task. For a thorough formal definition of a computational model of real-world time, we refer to [50, Section 3.1]. In what follows, we will argue why Assumption 1 is meaningful providing a strategy that achieves that asymptotic complexity – concretely, we will prove that our weak VDF is $(\mathcal{O}(\textbf{poly}(\ell)), \mathcal{O}(\ell \log \ell))-$sequential. Throughout this section, we will assume (time) complexity to be the number of arithmetic operations in $\mathbb{F}_p$, unless stated otherwise.

First note that there are many ways to compute an isogeny in this setting. In Subsection 3.2, we argued that this cannot be done efficiently by means of an equivalent smooth-norm ideal in the class group due to the size of p. Given that we work with a supersingular elliptic curve with unknown endomorphism ring, we can also not use a maximal order in $\mathrm{End}(E_0)$ as discussed in Subsection 3.3. Using classical modular polynomials is an option, but regardless of their efficiency, they only provide the codomain curve and do not allow us to evaluate the isogeny on points (see Subsection 3.4).

To the best of our knowledge, all other known ways of evaluating such isogenies require using its kernel polynomial in some way. This polynomial can be constructed from an ℓ-torsion kernel generator by means of Vélu-style formulae, or it can be found as a factor from the ℓ-division polynomial. The latter is a degree-$(\ell^2-1)/2$ polynomial which over $\mathbb{F}_p[x]$ factors into two irreducible polynomials of degree $(\ell-1)/2$ and $(\ell-1)/2$ irreducible polynomials of degree $\ell - 1$. These two factors of degree $(\ell-1)/2$ correspond exactly to the kernel polynomials of the horizontal isogenies, so the correct factors are easy to distinguish. Note that the ℓ-division polynomial is of degree $\mathcal{O}(\ell^2)$ however, such that it is infeasible to try to factor this in time $\mathcal{O}(\ell \log \ell)$. For a more elaborate argumentation of this statement, we refer to Appendix A. We will now discuss how to compute this kernel polynomial from a kernel generator, starting with the Fast Fourier Transform (FFT) for arithmetic in $\mathbb{F}_{p^k}$.

6.1 The Parallel FFT

Elements in $\mathbb{F}_{p^k}$ in our setting can be represented as polynomials modulo an irreducible polynomial of degree $k = \mathcal{O}(\ell)$. Hence, multiplying two elements in $\mathbb{F}_{p^k}$ is equivalent to multiplying two polynomials of degree $k-1$ over $\mathbb{F}_p$. The naive algorithm to multiply such polynomials requires $\mathcal{O}(\ell^2)$ $\mathbb{F}_p$-operations. However, it is possible to lower it down to $\mathcal{O}(\ell \log \ell)$ via the Fast Fourier transform (FFT) [18]. It is worth mentioning that this asymptotic complexity is theoretical and could be difficult to reach in practical applications. In what follows, we will uniquely discuss the best theoretical complexity ignoring engineering challenges. For the sake of designing a VDF, we are only interested in the best case scenario for our delay. In practice, given our choice of k, FFT may perform slightly worse.

One of the main advantages of FFT algorithms is that they can be parallelized. For instance, in [22], Cui-xiang, Guo-qiang and Ming-he describe a parallel FFT algorithm. Assuming one has access to arithmetic circuits of breadth m, this algorithm has complexity $\mathcal{O}((\ell/m)\log \ell)$ with a communication cost of $\mathcal{O}(\log m)$. In particular, for $\ell = m$, the complexity becomes $\mathcal{O}(\log \ell)$ and the communication cost becomes $\mathcal{O}(\log \ell)$. FFT can also be used for multiplying two elements in $\mathbb{F}_p$, but this speed-up is only asymptotic. In practice, even for p thousands of bits, the FFT does not outperform plain Montgomery multiplication. We refer the reader to [18] for further background on FFT.

Other algorithms for multiplying polynomials exist, such as the Toom-Cook multiplication [48,21]. To the best of our knowledge, none of these can be parallelized faster than the FFT. While addition of two polynomials can be done componentwise on all the coefficients with enough separate processors, we do not see how this can happen for multiplication.

6.2 Computing a Point of Order ℓ

An ℓ-torsion point is generated by sampling a random point and multiplying it by its cofactor; i.e. for each $P \in E_0(\mathbb{F}_{p^k})$ and $c = \#E(\mathbb{F}_{p^k})/\ell$ we have that $[c]P \in E_0[\ell]$. In practice, we can restrict ourselves to computing $x([c]P)$ since x-only arithmetic can be used to compute isogenies (see for instance [16]). This may require an isomorphism from $E_0/\mathbb{F}_p$ to a curve in Montgomery form, but this comes at negligible cost. Writing c as $\sum_{i=0}^{k-1} a_i p^i$, we can use the following strategy to obtain $[c]P$.

First, for all $i \in \{0, \ldots, k-1\}$, we compute and store $[a_i]P$. This can be done in parallel in $\mathcal{O}(\log p)$ $\mathbb{F}_{p^k}$-operations, which corresponds to $\mathcal{O}((\ell/m)\log p \log \ell)$

$\mathbb{F}_p$-operations using arithmetic circuits of breadth m for each of the arithmetic circuits of breadth $k = \mathcal{O}(\ell)$ we are using to compute the $[a_i]P$'s. This implies we should use arithmetic circuits of breadth mk just for this step.

We observe that

$$[c]P = [a_0]P + [p]([a_1]P + [p]([a_2]P + \ldots + [p]([a_{k-2}]P + [p][a_{k-1}]P))).$$

Since E is supersingular, $\pi^2 = [-p]$. Hence, to compute $[p]Q$ for any $Q \in E$, we need to apply the p-Frobenius twice. Each Frobenius costs $\mathcal{O}(\ell)$ $\mathbb{F}_p$-operations, which can be reduced to $\mathcal{O}(1)$ $\mathbb{F}_p$-multiplications using arithmetic circuits of breadth k. Summing two points $P_1, P_2 \in E(\mathbb{F}_{p^k})$ requires $\mathcal{O}(1)$ $\mathbb{F}_{p^k}$-operations, which amounts to $\mathcal{O}(\log \ell)$ $\mathbb{F}_p$-operations using parallel FFT with arithmetic circuits of breadth k. Therefore, each sum of the form $[a_{i-1}]P + [p][a_i]Q$ can be done in $\mathcal{O}(\log \ell)$ $\mathbb{F}_p$-operations using parallel FFT.

To compute $[c]P$, we need to perform $\mathcal{O}(\ell)$ operations of the form $[a_{i-1}]P + [p][a_i]Q$, which amounts to $\mathcal{O}(\ell \log \ell)$ $\mathbb{F}_p$-operations. Therefore, having arithmetic circuits of breadth mk, the asymptotic cost of computing a point of order ℓ is

$$\mathcal{O}(\max\{\ell \log \ell, (\ell/m) \log p \log \ell\})$$

$\mathbb{F}_p$-operations. Therefore, taking $m \approx \log p$, computing a point of order ℓ takes $\mathcal{O}(\ell \log \ell)$ $\mathbb{F}_p$-operations with arithmetic circuits of breadth $2k \log p$.

Remark 4. Note that in our weak VDF protocol one needs to sample two ℓ-torsion points corresponding to two horizontal ℓ-isogenies; one is on the curve itself and one is on the twist. In protocols such as CSIDH, this is typically done by using x-only arithmetic as described here, followed by a square check for the y-coordinate to see on which curve the point is. Given that a square check is much more expensive over $\mathbb{F}_{p^k}$ than over $\mathbb{F}_p$ for large k, one can instead opt to use the Elligator point sampling method (see for example [38,3]). Indeed, as our protocol does not need to differentiate between the ℓ-torsion point on the curve and the one on the twist, we can simply compute both simultaneously.

6.3 Computing the Kernel Polynomial

There are several ways of constructing the kernel polynomial given a kernel generator. For instance, in [5], they provide an asymptotic speed-up over the classical Vélu formulae by a square-root factor. In [30], a new algorithm to compute the kernel polynomials from irrational points is also provided. While these works may be of interest, they all assume the knowledge of the x-coordinate of an ℓ-torsion point, which we argued has already complexity $\mathcal{O}(\ell \log \ell)$. So it suffices for us to provide a way of computing the kernel polynomial in this time complexity.

Let $P \in E(\mathbb{F}_{p^{\ell-1}})$ be a point of order ℓ such that $x(P) \in \mathbb{F}_{p^k}$. Our goal is to compute the isogeny having kernel $\langle P \rangle$ only utilising the x-coordinate of P. We will show a strategy to do so having arithmetic circuits of breadth $\mathcal{O}(\ell)$. To obtain the set $\mathcal{P} \coloneqq \{x([s]P) \mid s = 1, \ldots, k\}$, using arithmetic circuits of breadth

$k = (\ell - 1)/2$, each of the arithmetic circuits of breadth k will compute one of the elements in $\mathcal{P}$ at the same time. The most demanding task is to compute $x([k]P)$, which requires $\mathcal{O}(\log \ell)$ $\mathbb{F}_{p^k}$-operations. Equivalently, $\mathcal{O}(\ell/m'(\log \ell)^2)$ $\mathbb{F}_p$-operations employing parallel FFT with arithmetic circuits of breadth m'. As a result, computing $\mathcal{P}$ takes $\mathcal{O}(\ell \log \ell)$ $\mathbb{F}_p$-operations using arithmetic circuits of breadth $m'k$.

The kernel polynomial is given precisely by

$$P(x) = \prod_{x_i \in \mathcal{P}} (x - x_i).$$

This product can be computed pairwise in a (binary) tree of height $\log(k)$, where each step requires some multiplications over $\mathbb{F}_{p^k}$. More precisely, using arithmetic circuits of breadth m', we can use parallel FFT such that it takes $\mathcal{O}((\ell/m') \log \ell)$ $\mathbb{F}_p$-operations. The computation of $P(x)$ will thus take $\mathcal{O}((\ell/m')(\log \ell)^2)$ $\mathbb{F}_p$-operations. Taking arithmetic circuits of breadth $m' = \lceil \log \ell \rceil$, we then have that computing this kernel polynomial requires at most $\mathcal{O}(\ell \log \ell)$ $\mathbb{F}_p$-operations. From this degree-k kernel polynomial $P(x) \in \mathbb{F}_p[x]$, one can evaluate the corresponding isogeny in time $\mathcal{O}(\ell)$ with well-known formulae such as those in [35].

7 Conclusion

In this paper, we have introduced a novel weak Verifiable Delay Function (VDF) that is resistant to quantum attacks. Our construction is based on isogenies, which are mappings between elliptic curves, and leverages the strengths of elliptic curves and elliptic products to enable efficient verification of slow one-dimensional isogenies. The slowness of these isogenies arises from the fact that their kernel generators are defined over large extension fields. This feature contributes to their resistance against quantum attacks. Our weak VDF incorporates two horizontal delay-generating isogenies, and their computation is verified in dimension two through the reconstruction of these isogenies.

We implemented the weak VDF described in Section 3 in SageMath. The source code is freely available at https://github.com/pq-vdf-isogeny/pq-vdf-isogeny. The purpose of this implementation is to demonstrate the correctness of the algorithm. It is important to note that this implementation should be considered as a proof-of-concept (the size of p in the default parameters is 256 bits and does not meet the security requirements), and there is room for optimizing several subroutines. Notably, the parallel algorithms outlined in Subsections 6.3 and 6.2 have not been included in the provided source code.

Additionally, to enhance performance, lower-level languages such as C and leveraging platform-specific instructions such as AVX could be utilized. By adopting these techniques, it is possible to significantly reduce the running time of the implementation. Ideally, when evaluating isogeny-based delay functions, the utilization of specialized hardware or Field-Programmable Gate Arrays (FPGAs) would be beneficial.

Throughout the paper, we have identified and discussed several open problems in this area. One such problem is the requirement for curves with an unknown endomorphism ring as input without the ability to rely on a trusted setup. This issue has been a persistent challenge in various isogeny-based protocols and continues to be an active area of research.

References

1. Arpin, S., Camacho-Navarro, C., Lauter, K., Lim, J., Nelson, K., Scholl, T., Sotáková, J.: Adventures in Supersingularland. Experimental Mathematics (2021), https://doi.org/10.1080/10586458.2021.1926009
2. 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
3. Banegas, G., Krämer, J., Lange, T., Meyer, M., Panny, L., Reijnders, K., Sotáková, J., Trimoska, M.: Disorientation faults in CSIDH. In: Advances in Cryptology – EUROCRYPT 2023, part II. pp. 310–342. Springer Nature Switzerland (2023)
4. Basso, A., Codogni, G., Connolly, D., De Feo, L., Fouotsa, T.B., Lido, G.M., Morrison, T., Panny, L., Patranabis, S., Wesolowski, B.: Supersingular curves you can trust. In: EUROCRYPT 2023, Part II. pp. 405–437. Springer (2023), https://doi.org/10.1007/978-3-031-30617-4_14
5. Bernstein, D.J., De Feo, L., Leroux, A., Smith, B.: Faster computation of isogenies of large prime degree. ANTS-XIV, The Open Book Series **4**(1), 39–55 (2020), https://doi.org/10.2140/obs.2020.4.39
6. Bernstein, D.J., Lange, T.: Montgomery curves and the Montgomery ladder. Cryptology ePrint Archive, Paper 2017/293 (2017), https://eprint.iacr.org/2017/293
7. Beullens, W., De Feo, L., Galbraith, S., Petit, C.: Proving knowledge of isogenies: a survey. Designs, Codes and Cryptography (2023), https://doi.org/10.1007/s10623-023-01243-3
8. Beullens, W., Kleinjung, T., Vercauteren, F.: CSI-FiSh: Efficient isogeny based signatures through class group computations. In: ASIACRYPT 2019, Part I. pp. 227–247. Springer (2019), https://doi.org/10.1007/978-3-030-34578-5_9
9. Boneh, D., Bonneau, J., Bünz, B., Fisch, B.: Verifiable delay functions. In: CRYPTO 2018, Part I. pp. 757–788. Springer (2018), https://doi.org/10.1007/978-3-319-96884-1_25
10. Boneh, D., Bünz, B., Fisch, B.: A survey of two verifiable delay functions. Cryptology ePrint Archive, Paper 2018/712 (2018), https://eprint.iacr.org/2018/712
11. Booher, J., Bowden, R., Doliskani, J., Fouotsa, T.B., Galbraith, S.D., Kunzweiler, S., Merz, S.P., Petit, C., Smith, B., Stange, K.E., Ti, Y.B., Vincent, C., Voloch, J.F., Weitkämper, C., Zobernig, L.: Failing to hash into supersingular isogeny graphs. Cryptology ePrint Archive, Report 2022/518 (2022), https://eprint.iacr.org/2022/518
12. Borodin, A., von zur Gathen, J., Hopcroft, J.: Fast parallel matrix and GCD computations. Information and Control **52**(3), 241–256 (1982), https://doi.org/10.1016/S0019-9958(82)90766-5
13. Cantor, D.G., Zassenhaus, H.: A new algorithm for factoring polynomials over finite fields. Mathematics of Computation **36**, 587–592 (1981), https://doi.org/10.2307/2007663

14. Castryck, W., Decru, T.: An efficient key recovery attack on SIDH. In: EUROCRYPT 2023, Part II. pp. 423–447. Springer (2023), https://doi.org/10.1007/978-3-031-30589-4_15
15. Castryck, W., Decru, T., Smith, B.: Hash functions from superspecial genus-2 curves using Richelot isogenies. Journal of Mathematical Cryptology **14**(1), 268–292 (2020), https://doi.org/10.1515/jmc-2019-0021
16. Castryck, W., Lange, T., Martindale, C., Panny, L., Renes, J.: CSIDH: an efficient post-quantum commutative group action. In: ASIACRYPT 2018, Part III. pp. 395–427. Springer (2018), https://doi.org/10.1007/978-3-030-03332-3_15
17. Chavez-Saab, J., Rodríguez-Henríquez, F., Tibouchi, M.: Verifiable isogeny walks: Towards an isogeny-based postquantum VDF. In: International Conference on Selected Areas in Cryptography. pp. 441–460. Springer (2022), https://doi.org/10.1007/978-3-030-99277-4_21
18. Chu, E., George, A.: Inside the FFT black box: serial and parallel fast Fourier transform algorithms. CRC press (1999), https://doi.org/10.1201/9780367802332
19. Cohen, H., Frey, G., Avanzi, R., Doche, C., Lange, T., Nguyen, K., Vercauteren, F.: Handbook of elliptic and hyperelliptic curve cryptography. CRC press (2005)
20. Cong, K., Lai, Y.F., Levin, S.: Efficient isogeny proofs using generic techniques. In: Applied Cryptography and Network Security. pp. 248–275. Springer Nature Switzerland (2023), https://doi.org/10.1007/978-3-031-33491-7_10
21. Cook, S.A.: On the minimum computation time of functions. Ph.D. thesis, Harvard University (1966)
22. Cui-xiang, Z., Guo-qiang, H., Ming-he, H.: Some new parallel fast Fourier transform algorithms. In: Sixth International Conference on Parallel and Distributed Computing Applications and Technologies (PDCAT'05). pp. 624–628 (2005), https://doi.org/10.1109/PDCAT.2005.224
23. De Feo, L.: Mathematics of isogeny based cryptography. The Arxive abs/1711.04062 (2017), http://arxiv.org/abs/1711.04062
24. 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), https://doi.org/10.1515/jmc-2012-0015
25. De Feo, L., Masson, S., Petit, C., Sanso, A.: Verifiable delay functions from supersingular isogenies and pairings. In: ASIACRYPT 2019, Part I. pp. 248–277. Springer (2019), https://doi.org/10.1007/978-3-030-34578-5_10
26. Delfs, C., Galbraith, S.D.: Computing isogenies between supersingular elliptic curves over $\mathbb{F}_p$. Designs, Codes and Cryptography **78**(2), 425–440 (2016), https://doi.org/10.1007/s10623-014-0010-1
27. Doliskani, J.: Toward an optimal quantum algorithm for polynomial factorization over finite fields. Quantum Info. Comput. **19**(1–2), 1–13 (2019)
28. Döttling, N., Garg, S., Malavolta, G., Vasudevan, P.N.: Tight verifiable delay functions. In: Security and Cryptography for Networks. pp. 65–84. Springer (2020), https://doi.org/10.1007/978-3-030-57990-6_4
29. Elkies, N.D.: Elliptic and modular curves over finite fields and related computational issues. In: Computational perspectives on number theory, Studies in Advanced Mathematics 7. pp. 21–76. AMS (1998)
30. Eriksen, J.K., Panny, L., Sotáková, J., Veroni, M.: Deuring for the people: Supersingular elliptic curves with prescribed endomorphism ring in general characteristic. Cryptology ePrint Archive, Paper 2023/106 (2023), https://eprint.iacr.org/2023/106
31. Fouotsa, T.B., Petit, C.: A new adaptive attack on SIDH. In: CT-RSA 2022. pp. 322–344. Springer (2022), https://doi.org/10.1007/978-3-030-95312-6_14

32. von zur Gathen, J.: Parallel algorithms for algebraic problems. In: Symposium on Theory of Computing. p. 17–23. STOC '83, Association for Computing Machinery (1983), https://doi.org/10.1145/800061.808728
33. Giacomelli, I., Madsen, J., Orlandi, C.: ZKBoo: Faster zero-knowledge for boolean circuits. In: Conference on Security Symposium. p. 1069–1083. SEC'16, USENIX Association (2016)
34. Kani, E.: The number of curves of genus two with elliptic differentials. Journal für die reine und angewandte Mathematik **1997**(485), 93–122 (1997), https://doi.org/10.1515/crll.1997.485.93
35. Kohel, D.R.: Endomorphism rings of elliptic curves over finite fields. Ph.D. thesis, University of California, Berkeley (1996)
36. Leroux, A.: A new isogeny representation and applications to cryptography. In: ASIACRYPT 2022, Part II. pp. 3–35. Springer (2022), https://doi.org/10.1007/978-3-031-22966-4_1
37. Maino, L., Martindale, C., Panny, L., Pope, G., Wesolowski, B.: A direct key recovery attack on SIDH. In: EUROCRYPT 2023, Part II. pp. 448–471. Springer (2023), https://doi.org/10.1007/978-3-031-30589-4_16
38. Meyer, M., Campos, F., Reith, S.: On lions and elligators: An efficient constant-time implementation of CSIDH. In: Post-Quantum Cryptography 2019. pp. 307–325. Springer (2019), https://doi.org/10.1007/978-3-030-25510-7_17
39. Morgenstern, M., Shamir, E.: Parallel algorithms for arithmetics, irreducibility and factoring of GFq-polynomials. Tech. rep., Stanford University (1983), https://dl.acm.org/doi/10.5555/892306
40. Mula, M., Murru, N., Pintore, F.: On Random Sampling of Supersingular Elliptic Curves. Cryptology ePrint Archive, Paper 2022/528 (2022), https://eprint.iacr.org/2022/528
41. Panny, L.: CSI-FiSh really isn't polynomial-time, https://yx7.cc/blah/2023-04-14.html#fn5
42. Pietrzak, K.: Simple verifiable delay functions. In: Innovations in Theoretical Computer Science (ITCS 2019). Leibniz International Proceedings in Informatics (LIPIcs), vol. 124, pp. 60:1–60:15. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik (2018), https://doi.org/10.4230/LIPIcs.ITCS.2019.60
43. Robert, D.: Breaking SIDH in polynomial time. In: EUROCRYPT 2023, Part II. pp. 472–503. Springer (2023), https://doi.org/10.1007/978-3-031-30589-4_17
44. Silverman, J.H.: The arithmetic of elliptic curves, vol. 106. Springer (2009), https://doi.org/10.1007/978-0-387-09494-6
45. Smith, B.: Explicit endomorphisms and correspondences. Ph.D. thesis, University of Sydney (2006)
46. Sutherland, A.: On the evaluation of modular polynomials. The Open Book Series **1**(1), 531–555 (2013), https://dx.doi.org/10.2140/obs.2013.1.531
47. Tan, T.G., Sharma, V., Li, Z., Szalachowski, P., Zhou, J.: ZKBdf: A ZKBoo-based quantum-secure verifiable delay function with prover-secret. Cryptology ePrint Archive, Paper 2022/1373 (2022), https://eprint.iacr.org/2022/1373
48. Toom, A.L.: The complexity of a scheme of functional elements realizing the multiplication of integers. In: Soviet Mathematics Doklady. vol. 3, pp. 714–716 (1963)
49. von zur Gathen, J., Panario, D.: Factoring polynomials over finite fields: A survey. Journal of Symbolic Computation **31**(1), 3–17 (2001), https://doi.org/10.1006/jsco.1999.1002
50. Wesolowski, B.: Efficient verifiable delay functions. Journal of Cryptology **33**(4), 2113–2147 (2020), https://doi.org/10.1007/s00145-020-09364-x

A Factoring the ℓ-division Polynomial

The ℓ-division polynomial in our setting is a degree-$(\ell^2-1)/2$ polynomial in $\mathbb{F}_p[x]$. The fastest way to construct this polynomial is by a recurrence relationship taking $\mathcal{O}(\ell^2 \log \ell)$ multiplications. Remark that the ℓ-division polynomial can be precomputed from a specific form of elliptic curves (e.g. based on a Montgomery coefficient A). Both the degree of the ℓ-division polynomial as well as the degree of A in this precomputation are $\mathcal{O}(\ell^2)$. Hence, using arithmetic circuits of breadth $m = \mathcal{O}(\ell^4)$, one can evaluate the expression in A by means of square-and-multiply in time $\mathcal{O}(\log \ell)$.

The ℓ-division polynomial factors in $\ell+1$ factors of degree $(\ell-1)/2$ in $\mathbb{F}_{p^2}[x]$, where each factor determines a kernel polynomial of an ℓ-isogeny. In [49], von zur Gathen and Panario survey some algorithms to factor polynomials over finite field. Even though in our case we could use the more efficient equal-degree factorization algorithms, their complexity is not competitive with the strategy we described in Section 6. However, it is worth noting that this survey does not consider parallel versions of these algorithms. In [32], Gathen describes a parallel version of the Cantor-Zassenhaus's algorithm [13]. Adapting the complexity in [32, Theorem 4.1] to our setting, factoring the ℓ-division over $\mathbb{F}_{p^2}$ requires $\mathcal{O}(\log^2 \ell \log p)$ $\mathbb{F}_p$-operations utilising arithmetic circuits of breadth $\mathcal{O}(\mathbf{poly}(\ell^2))$.

Despite being polynomial in ℓ^2, and in turn allowed by the definition of weak VDF, the exponent of $\mathbf{poly}(\ell^2)$ is likely to be huge. For instance, one of the steps of Cantor-Zassenhaus parallel algorithm relies on the computation of the quotient and reminder of two polynomials. As explained in [12, Remark 2], this step itself reaches complexity $\mathcal{O}(\log^2 n)$ when $\mathcal{O}(n^{3.5})$ parallel processors are employed, where n is the degree of the two polynomials.[6] This essential step is required for $\mathbf{poly}(n)$ parallel steps, further increasing the breadth of arithmetic circuits required by Cantor-Zassenhaus parallel algorithm. A brief discussion on the exponent of the polynomial describing the breadth of arithmetic circuits required by this algorithm is contained in [39], where the authors estimate the exponent to be 13. Thus, one would need arithmetic circuits of breadth $\mathcal{O}(\ell^{26})$ to apply this algorithm in our case. For instance, if the delay parameter t provided as input in Setup is as small as 2^5, one already needs arithmetic circuits of breadth $\sim 2^{130}$, which is an unrealistic requirement. On top of this analysis, we shall also mention that the algorithm is theoretical and does not take into account communication costs. A real-world implementation of this algorithm would be a major breakthrough on its own.

Finally, to the best of our knowledge, no known quantum algorithm can help us factor polynomials over finite fields faster. Doliskani gives a quantum algorithm that can factor a degree-n polynomial over $\mathbb{F}_q$ in $\mathcal{O}(n^{1+o(1)} \log^{2+o(1)} q)$ bit operations [27]. In our case however, this reduces to $\mathcal{O}(\ell^{2+o(1)} \log^{2+o(1)} p)$ for factoring the ℓ-division polynomial, and hence provides no speed-up.

[6] It is possible to have complexity $\mathcal{O}(\log n)$ if arithmetic circuits of breadth $\mathcal{O}(n^{15})$ are used.