Verifiable Delay Functions from Isogenies and Pairings

Luca De Feo

joint work with J. Burdges, S. Masson, C. Petit, A. Sanso

Université Paris Saclay – UVSQ, France

July 13, 2019, SIAM AG, Bern

Slides online at `https://defeo.lu/docet`

Tired of *SIDH?

Tired of *SIDH?

Enough quantum FUD?

Tired of *SIDH?

Enough quantum FUD?

Ready for a new buzzword?


#BLOCKCHAIN

Distributed lottery

Participants **A, B, . . . , Z** want to agree on a random winning ticket.

Flawed protocol

- Each participant x broadcasts a random string s_x;
- Winning ticket is $H(s_A, \ldots, s_b)$.

Distributed lottery

Participants **A, B, . . . , Z** want to agree on a random winning ticket.

Flawed protocol

- Each participant x broadcasts a random string s_x;
- Winning ticket is $H(s_A, \dots, s_b)$.

Fixes

- Make the hash function **slooooooooooooooooooooooooooooooow**;

Distributed lottery

Participants **A, B, . . . , Z** want to agree on a random winning ticket.

Flawed protocol

- Each participant x broadcasts a random string s_x;
- Winning ticket is $H(s_A, \dots, s_b)$.

Fixes

- Make the hash function **slooooooooooooooooooooooooooooow**;
- Make it possible to verify $w = H(s_A, \dots, s_b)$ **fast**.

Verifiable Delay Functions

Wanted

Function (family) $f : X \to Y$ s.t.:

- Evaluating $f(x)$ takes long time:
  - uniformly long time,
  - on almost all random inputs x,
  - even after having seen many values of $f(x')$,
  - even given massive number of processors;
- Verifying $y = f(x)$ is efficient:
  - ideally, exponential separation between evaluation and verification.

Verifiable Delay Functions

Wanted

Function (family) $f : X \to Y$ s.t.:

- Evaluating $f(x)$ takes long time:
  - uniformly long time,
  - on almost all random inputs x,
  - even after having seen many values of $f(x')$,
  - even given massive number of processors;
- Verifying $y = f(x)$ is efficient:
  - ideally, exponential separation between evaluation and verification.

Exercise

Verifiable Delay Functions

Wanted

Function (family) $f : X \to Y$ s.t.:

- Evaluating $f(x)$ takes long time:
  - uniformly long time,
  - on almost all random inputs x,
  - even after having seen many values of $f(x')$,
  - even given massive number of processors;
- Verifying $y = f(x)$ is efficient:
  - ideally, exponential separation between evaluation and verification.

Exercise

Think of a function you like with these properties

Verifiable Delay Functions

Wanted

Function (family) $f : X \rightarrow Y$ s.t.:

- Evaluating $f(x)$ takes long time:
  - uniformly long time,
  - on almost all random inputs x,
  - even after having seen many values of $f(x')$,
  - even given massive number of processors;
- Verifying $y = f(x)$ is efficient:
  - ideally, exponential separation between evaluation and verification.

Exercise

Think of a function you like with these properties

Got it?

Verifiable Delay Functions

Wanted

Function (family) $f : X \to Y$ s.t.:

- Evaluating $f(x)$ takes long time:
  - uniformly long time,
  - on almost all random inputs x,
  - even after having seen many values of $f(x')$,
  - even given massive number of processors;
- Verifying $y = f(x)$ is efficient:
  - ideally, exponential separation between evaluation and verification.

Exercise

Think of a function you like with these properties

Got it?

You’re probably wrong!

Sequentiality

Ideal functionality:

$$y = f(x) = \underbrace{H(H(\cdots(H(x))))}_{T \text{ times}}$$

- Sequential assuming hash output "unpredictability",
- but how do you verify?

VDFs from groups of unknown order

Setup

A group of unknown order, e.g.:

- $\mathbb{Z}/N\mathbb{Z}$ with $N = pq$ an RSA modulus, p, q unknown (e.g., generated by some trusted authority),
- Class group of imaginary quadratic order.

Evaluation

With delay parameter T:

$$\begin{aligned} f : G &\longrightarrow G \\ x &\longmapsto x^{2^T} \end{aligned}$$

Conjecturally, fastest algorithm is repeated squaring.

Verification (Wesolowski 2019, Pietrzak 2019)

VDFs from groups of unknown order

Setup

A group of unknown order, e.g.:

- $\mathbb{Z}/N\mathbb{Z}$ with $N = pq$ an RSA modulus, p, q unknown (e.g., generated by some trusted authority),
- Class group of imaginary quadratic order.

Evaluation

With delay parameter T:

$$\begin{aligned} f : G &\longrightarrow G \\ x &\longmapsto x^{2^T} \end{aligned}$$

Conjecturally, fastest algorithm is repeated squaring.

Verification (Wesolowski 2019, Pietrzak 2019)

Aha!

Isogeny <3 Pairing

Let $\phi : E \to E'$, let $P \in E[N]$ and $Q \in E'[N]$. Then

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

$$\begin{tikzcd}
X_1 \times X_2 \arrow[r, "\phi \times 1"] \arrow[d, "1 \times \hat{\phi}"'] & X_1 \times X_2 \arrow[d, "e_N"] \\
X_1 \times X_2 \arrow[r, "e_N"'] & \mathbb{F}_{p^k}
\end{tikzcd}$$

Isogeny <3 Pairing

Let $\phi : E \to E'$, let $P \in E[N]$ and $Q \in E'[N]$. Then

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

$$\begin{tikzcd}
X_1 \times X_2 \arrow[r, "\phi \times 1"] \arrow[d, "1 \times \hat{\phi}"'] & X_1 \times X_2 \arrow[d, "e_N"] \\
X_1 \times X_2 \arrow[r, "e_N"'] & \mathbb{F}_{p^k}
\end{tikzcd}$$

Idea #1

Use the equation for a BLS-like signature scheme: US patent 8,250,367 (Broker, Charles, Lauter).

Isogeny VDF

Assume $\deg \phi = 2^T$

$$e_N(\phi(P), \phi(Q)) = e_N(P, Q)^{2^T}$$

Right side: known group structure: $2^T \quad \to \quad 2^T \bmod p^k - 1$;
Left side: can evaluate ϕ in less than T steps?

Isogeny VDF ($\mathbb{F}_p$-version)

Setup

- Pairing friendly supersingular curve $E/\mathbb{F}_p$
- Isogeny $\phi : E \to E'$ of degree 2^T,
- Point $P \in E[(N, \pi - 1)]$, image $\phi(P)$.

Evaluation

Input: random $Q \in E'[(N, \pi + 1)]$,

Output: $\hat{\phi}(Q)$.

Verification

$$e_N(P, \hat{\phi}(Q)) \quad \stackrel{?}{=} \quad e_N(\phi(P), Q).$$

Isogeny VDF ($\mathbb{F}_p$-version)

Trusted Setup

- Pairing friendly supersingular curve $E/\mathbb{F}_p$ **with unknown endomorphism ring!!!**
- Isogeny $\phi : E \to E'$ of degree 2^T,
- Point $P \in E[(N, \pi - 1)]$, image $\phi(P)$.

Evaluation

Input: random $Q \in E'[(N, \pi + 1)]$,

Output: $\hat{\phi}(Q)$.

Verification

$$e_N(P, \hat{\phi}(Q)) \quad \stackrel{?}{=} \quad e_N(\phi(P), Q).$$

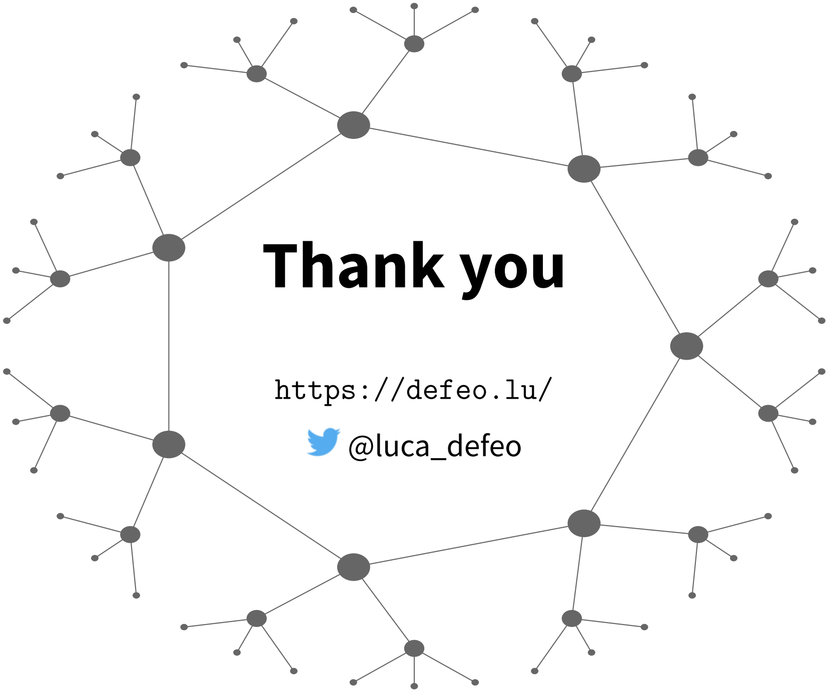
Sequentiality?

Wesolowski, Pietrzak: $x \longmapsto x^2$

Isogenies: $x \longmapsto x\dfrac{x\alpha_i - 1}{x - \alpha_i}$

No speedup? Even with unlimited parallelism? Really?

See Bernstein, Sorenson. Modular exponentiation via the explicit Chinese remainder theorem.



Thank you

`https://defeo.lu/`

@luca_defeo