Efficient supersingularity testing over $\mathbb{F}_p$ and CSIDH key validation

Gustavo Banegas [1], **Valerie Gilchrist** [2,1], Benjamin Smith [1]

[1]Inria and Laboratoire d'Informatique de l'Ecole polytechnique, Institut Polytechnique de Paris, Palaiseau, France

[2]University of Waterloo, Canada

September 22, 2022

Supersingular elliptic curves and CSIDH key validation

CSIDH is an isogeny-based cryptosystem that meets post-quantum security requirements.
It's a non-interactive key exchange scheme so key-validation is essential.

Supersingular elliptic curves and CSIDH key validation

CSIDH is an isogeny-based cryptosystem that meets post-quantum security requirements. It's a non-interactive key exchange scheme so key-validation is essential.

What is the CSIDH public key?

$$A \in \mathbb{F}_p \text{ such that } \mathcal{E}_A : y^2 = x(x^2 + Ax + 1) \text{ s.t. } \mathcal{E}_A \text{ is supersingular}$$

i.e. $\mathcal{E}_A(\mathbb{F}_p)$ has $p + 1$ points

Supersingular elliptic curves and CSIDH key validation

CSIDH is an isogeny-based cryptosystem that meets post-quantum security requirements. It's a non-interactive key exchange scheme so key-validation is essential.

What is the CSIDH public key?

$$A \in \mathbb{F}_p \text{ such that } \mathcal{E}_A : y^2 = x(x^2 + Ax + 1) \text{ s.t. } \mathcal{E}_A \text{ is supersingular}$$

i.e. $\mathcal{E}_A(\mathbb{F}_p)$ has $p+1$ points

Problem: how should we check if $\mathcal{E}_A$ is supersingular?

Random Point Test

- It seeks to find a point of order $p+1$;

Random Point Test

- It seeks to find a point of order $p + 1$;
- Hasse's theorem $\implies$ sufficient to find a point of order $N \geq 4\sqrt{p}$ s.t. $N \mid p + 1$

Random Point Test

- It seeks to find a point of order $p+1$;
- Hasse's theorem $\implies$ sufficient to find a point of order $N \geq 4\sqrt{p}$ s.t. $N \mid p+1$
- The factorization of $p+1$ must be known (as in the CSIDH setting)

    In CSIDH,

$$p+1 = 4\prod_{i=1}^{n} \ell_i$$

Random Point Test

- It seeks to find a point of order $p + 1$;
- Hasse's theorem $\implies$ sufficient to find a point of order $N \geq 4\sqrt{p}$ s.t. $N \mid p + 1$
- The factorization of $p + 1$ must be known (as in the CSIDH setting)

  In CSIDH,

$$p + 1 = 4 \prod_{i=1}^{n} \ell_i$$

Algorithm: Sample point P from $\mathcal{E}_A(\mathbb{F}_p)$, compute $Q_i = [(p + 1)/\ell_i]P$ for several prime divisors ℓ_i of $p + 1$

Random Point Test

- It seeks to find a point of order $p+1$;
- Hasse's theorem $\implies$ sufficient to find a point of order $N \geq 4\sqrt{p}$ s.t. $N \mid p+1$
- The factorization of $p+1$ must be known (as in the CSIDH setting)

  In CSIDH,

$$p+1 = 4\prod_{i=1}^{n} \ell_i$$

Algorithm: Sample point P from $\mathcal{E}_A(\mathbb{F}_p)$, compute $Q_i = [(p+1)/\ell_i]P$ for several prime divisors ℓ_i of $p+1$

- if $Q_i \neq 0$ and $[\ell_i]Q_i = 0$ then ℓ_i is a divisor of N;

Random Point Test

- It seeks to find a point of order $p+1$;
- Hasse's theorem $\implies$ sufficient to find a point of order $N \geq 4\sqrt{p}$ s.t. $N \mid p+1$
- The factorization of $p+1$ must be known (as in the CSIDH setting)

  In CSIDH,

$$p+1 = 4\prod_{i=1}^{n} \ell_i$$

Algorithm: Sample point P from $\mathcal{E}_A(\mathbb{F}_p)$, compute $Q_i = [(p+1)/\ell_i]P$ for several prime divisors ℓ_i of $p+1$

- if $Q_i \neq 0$ and $[\ell_i]Q_i = 0$ then ℓ_i is a divisor of N;
- if $[\ell_i]Q_i \neq 0$, then the curve is not supersingular

Random Point Test

- It seeks to find a point of order $p+1$;
- Hasse's theorem $\implies$ sufficient to find a point of order $N \geq 4\sqrt{p}$ s.t. $N \mid p+1$
- The factorization of $p+1$ must be known (as in the CSIDH setting)

  In CSIDH,

$$p+1 = 4\prod_{i=1}^{n} \ell_i$$

Algorithm: Sample point P from $\mathcal{E}_A(\mathbb{F}_p)$, compute $Q_i = [(p+1)/\ell_i]P$ for several prime divisors ℓ_i of $p+1$

- if $Q_i \neq 0$ and $[\ell_i]Q_i = 0$ then ℓ_i is a divisor of N;
- if $[\ell_i]Q_i \neq 0$, then the curve is not supersingular
- repeat for enough ℓ_i such that their product exceeds $4\sqrt{p}$;

Random Point Test

- It seeks to find a point of order $p+1$;
- Hasse's theorem $\implies$ sufficient to find a point of order $N \geq 4\sqrt{p}$ s.t. $N \mid p+1$
- The factorization of $p+1$ must be known (as in the CSIDH setting)

  In CSIDH,

$$p+1 = 4\prod_{i=1}^{n} \ell_i$$

Algorithm: Sample point P from $\mathcal{E}_A(\mathbb{F}_p)$, compute $Q_i = [(p+1)/\ell_i]P$ for several prime divisors ℓ_i of $p+1$

- if $Q_i \neq 0$ and $[\ell_i]Q_i = 0$ then ℓ_i is a divisor of N;
- if $[\ell_i]Q_i \neq 0$, then the curve is not supersingular
- repeat for enough ℓ_i such that their product exceeds $4\sqrt{p}$;

For CSIDH-512, where $n = 74$, this results in ≈ 33 scalar multiplications.

Product Tree Test

Product tree version (currently in-use in CSIDH/CTIDH)

- The same small primes are being used in the computation of the Q_i so compute using a product tree;

Product Tree Test

Product tree version (currently in-use in CSIDH/CTIDH)

- The same small primes are being used in the computation of the Q_i so compute using a product tree;
- It searches tree depth first;

Product Tree Test

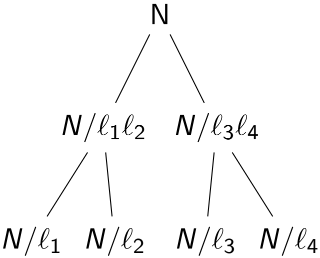
Product tree version (currently in-use in CSIDH/CTIDH)

- The same small primes are being used in the computation of the Q_i so compute using a product tree;
- It searches tree depth first;
- It has a faster run-time but it uses more memory.

Product Tree Test

Product tree version (currently in-use in CSIDH/CTIDH)

- The same small primes are being used in the computation of the Q_i so compute using a product tree;
- It searches tree depth first;
- It has a faster run-time but it uses more memory.



Isogeny Background

An *isogeny* is a non-constant, rational map between two elliptic curves.

Isogeny Background

An *isogeny* is a non-constant, rational map between two elliptic curves.

An ℓ*-isogeny graph*:

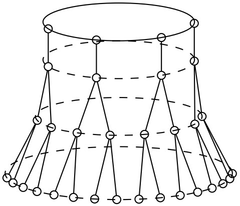
- vertices = ℓ-isogenous curves (up to isomorphism)
- edges = ℓ-isogenies (up to isomorphism)

Sutherland Test

2-isogeny graphs over $\mathbb{F}_p$ form a forest of same-sized trees, where the roots are connected by a cycle. We call them *volcanoes*.

The cycle of roots is the *crater*.

The leaves of the trees form the *floor*.

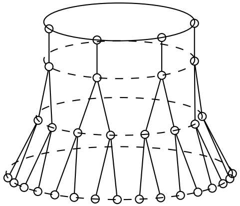


Sutherland Test

2-isogeny graphs over $\mathbb{F}_p$ form a forest of same-sized trees, where the roots are connected by a cycle. We call them *volcanoes*.

The cycle of roots is the *crater*.

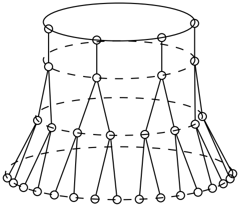
The leaves of the trees form the *floor*.



Supersingular 2-isogeny graphs over $\mathbb{F}_{p^2}$ will form a 3-regular, connected graph.

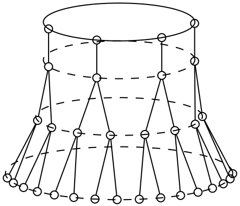
Ordinary (not supersingular) graphs over $\mathbb{F}_{p^2}$ form a larger volcano.

Sutherland Test



Sutherland's test (2011) aims to determine whether a given curve is supersingular or ordinary by identifying its 2-isogeny graph over $\mathbb{F}_{p^2}$ as either a volcano or a 3-regular graph.

Sutherland Test

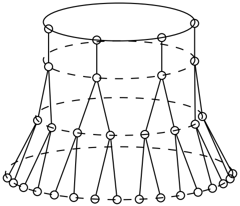


Sutherland's test (2011) aims to determine whether a given curve is supersingular or ordinary by identifying its 2-isogeny graph over $\mathbb{F}_{p^2}$ as either a volcano or a 3-regular graph.

It does so by

- beginning three (non-backtracking) paths in the $\mathbb{F}_{p^2}$ 2-isogeny graph of the curve;

Sutherland Test

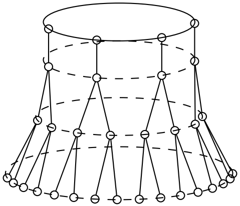


Sutherland's test (2011) aims to determine whether a given curve is supersingular or ordinary by identifying its 2-isogeny graph over $\mathbb{F}_{p^2}$ as either a volcano or a 3-regular graph.

It does so by

- beginning three (non-backtracking) paths in the $\mathbb{F}_{p^2}$ 2-isogeny graph of the curve;
- stepping through each path $\log_2 p + 1$ times (the max. height of the ordinary volcano);

Sutherland Test



Sutherland's test (2011) aims to determine whether a given curve is supersingular or ordinary by identifying its 2-isogeny graph over $\mathbb{F}_{p^2}$ as either a volcano or a 3-regular graph.

It does so by

- beginning three (non-backtracking) paths in the $\mathbb{F}_{p^2}$ 2-isogeny graph of the curve;
- stepping through each path $\log_2 p + 1$ times (the max. height of the ordinary volcano);
- if none of the paths hit the floor, then the graph is not a volcano.

Sutherland Test

We will traverse the graph using *modular polynomials*.
The (classical) modular polynomial of level 2 is

$$\Phi_2(j_1, j_2) = j_1^3 + j_2^3 - j_1^2 j_2^2 + 1488(j_1^2 j_2 + j_1 j_2^2) - 162000(j_1^2 + j_2^2) + 40773375 j_1 j_2 + 8748000000(j_1 + j_2) - 157464000000000\,.$$

Sutherland Test

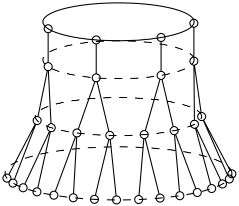
We will traverse the graph using *modular polynomials*.
The (classical) modular polynomial of level 2 is

$$\Phi_2(j_1, j_2) = j_1^3 + j_2^3 - j_1^2 j_2^2 + 1488(j_1^2 j_2 + j_1 j_2^2) - 162000(j_1^2 + j_2^2) \\ + 40773375 j_1 j_2 + 8748000000(j_1 + j_2) - 157464000000000\,.$$

It has the property that

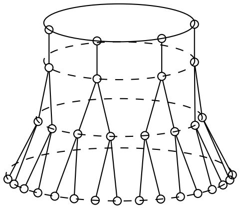
$$\text{there exists a 2-isogeny } \mathcal{E}_1 \to \mathcal{E}_2 \iff \Phi_2(j(\mathcal{E}_1), j(\mathcal{E}_2)) = 0\,.$$

Sutherland Test



Compute and factor the cubic $\Phi_2(X, j(\mathcal{E}_0)) = 0$.

Sutherland Test



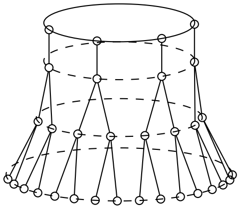
Compute and factor the cubic $\Phi_2(X, j(\mathcal{E}_0)) = 0$.

...

Compute and factor the quadratic $\Phi_2(X, j(\mathcal{E}_i))/(X - j(\mathcal{E}_{i-1})) = 0$.

- if it is irreducible, we have hit a leaf, so we are ordinary

Sutherland Test



Compute and factor the cubic $\Phi_2(X, j(\mathcal{E}_0)) = 0$.

...

Compute and factor the quadratic $\Phi_2(X, j(\mathcal{E}_i))/(X - j(\mathcal{E}_{i-1})) = 0$.

- if it is irreducible, we have hit a leaf, so we are ordinary

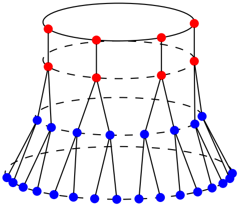
Repeat $\log_2 p + 1$ times.

- after this many steps, we have walked the maximal height of the volcano, so we must be supersingular

Modified Sutherland

Sutherland's modification in the $\mathbb{F}_p$ case:

- Assuming we are supersingular, the $\mathbb{F}_p$ volcano will be very short—only two levels tall;
- Within two steps, one neighbour will be defined over $\mathbb{F}_{p^2}$. We take this path down;
- Gives an approximate $3\times$ speedup.



Modified Sutherland

Our modifications:

- Improved bound on the maximum length of the descending path (if ordinary) of the volcano

$$\log_2 p + 1 \mapsto \frac{1}{2}\log_2 p + 1;$$

Modified Sutherland

Our modifications:

- Improved bound on the maximum length of the descending path (if ordinary) of the volcano

$$\log_2 p + 1 \mapsto \frac{1}{2}\log_2 p + 1;$$

- We replaced the modular polynomial computations by computing explicit 2-isogenies as follows:

$$\varphi : \mathcal{E} \longrightarrow \mathcal{E}/\langle(\alpha, 0)\rangle \cong \mathcal{E}' : y^2 = x(x^2 + a_2'x + a_4').$$

Doliskani Test

For $m \geq 0$, the m-th division polynomial $\psi_{\mathcal{E},m}$ of an elliptic curve $\mathcal{E}$ satisfies

$$\psi_{\mathcal{E},m}(x(P), y(P)) = 0 \iff P \in \mathcal{E}[m] \setminus \{0\}\,.$$

Doliskani Test

For $m \geq 0$, the m-th division polynomial $\psi_{\mathcal{E},m}$ of an elliptic curve $\mathcal{E}$ satisfies

$$\psi_{\mathcal{E},m}(x(P), y(P)) = 0 \iff P \in \mathcal{E}[m] \setminus \{0\}\,.$$

We also have that

$$\psi_{\mathcal{E},p}(x)^2 = 1 \iff \mathcal{E} \text{ is supersingular.}$$

(Note: a curve is supersingular $\iff$ it has no p-torsion over any field extension)

Doliskani Test

For $m \geq 0$, the m-th division polynomial $\psi_{\mathcal{E},m}$ of an elliptic curve $\mathcal{E}$ satisfies

$$\psi_{\mathcal{E},m}(x(P), y(P)) = 0 \iff P \in \mathcal{E}[m] \setminus \{0\}.$$

We also have that

$$\psi_{\mathcal{E},p}(x)^2 = 1 \iff \mathcal{E} \text{ is supersingular.}$$

(Note: a curve is supersingular $\iff$ it has no p-torsion over any field extension)

Doliskani's test (2018) applies basic Polynomial Identity Testing to check this criterion.

Doliskani Test

For $m \geq 0$, the m-th division polynomial $\psi_{\mathcal{E},m}$ of an elliptic curve $\mathcal{E}$ satisfies

$$\psi_{\mathcal{E},m}(x(P), y(P)) = 0 \iff P \in \mathcal{E}[m] \setminus \{0\}\,.$$

We also have that

$$\psi_{\mathcal{E},p}(x)^2 = 1 \iff \mathcal{E} \text{ is supersingular.}$$

(Note: a curve is supersingular $\iff$ it has no p-torsion over any field extension)

Doliskani's test (2018) applies basic Polynomial Identity Testing to check this criterion.
For a random $u \in \mathbb{F}_{p^2}$,

$$\psi_{\mathcal{E},p}(u)^2 = 1 \underset{\text{prob. 1-1/2p}}{\Longrightarrow} \psi^2_{\mathcal{E},p} = 1 \iff \mathcal{E} \text{ s.s.}$$

Modified Doliskani

Our modification:

Scalar multiplication can be defined in terms of division polynomials as

$$[m](x, y) = \left(\frac{\phi_{\mathcal{E},m}(x)}{\psi_{\mathcal{E},m}(x)^2}, \frac{\omega_{\mathcal{E},m}(x, y)}{\psi_{\mathcal{E},m}(x)^3} \right)$$

where $\phi_{\mathcal{E},m}(x)$ and $\omega_{\mathcal{E},m}(x, y)$ rely on $\psi_{\mathcal{E},m}(x)$.

Modified Doliskani

Our modification:
Scalar multiplication can be defined in terms of division polynomials as

$$[m](x, y) = \left(\frac{\phi_{\mathcal{E},m}(x)}{\psi_{\mathcal{E},m}(x)^2}, \frac{\omega_{\mathcal{E},m}(x, y)}{\psi_{\mathcal{E},m}(x)^3} \right)$$

where $\phi_{\mathcal{E},m}(x)$ and $\omega_{\mathcal{E},m}(x, y)$ rely on $\psi_{\mathcal{E},m}(x)$.

This tells us that if $(X : Y : Z) = [p](x, y)$, then $X = \lambda\phi_p(u)$ and $Z = \lambda\psi_p^2(u)$ where λ is a common projective factor.

Modified Doliskani

Our modification:
Scalar multiplication can be defined in terms of division polynomials as

$$[m](x,y) = \left(\frac{\phi_{\mathcal{E},m}(x)}{\psi_{\mathcal{E},m}(x)^2}, \frac{\omega_{\mathcal{E},m}(x,y)}{\psi_{\mathcal{E},m}(x)^3} \right)$$

where $\phi_{\mathcal{E},m}(x)$ and $\omega_{\mathcal{E},m}(x,y)$ rely on $\psi_{\mathcal{E},m}(x)$.

This tells us that if $(X:Y:Z) = [p](x,y)$, then $X = \lambda\phi_p(u)$ and $Z = \lambda\psi_p^2(u)$ where λ is a common projective factor.

In our version of Doliskani's test, we use (projective) differential addition to compute

$$[p](u:1) = \left(\lambda\phi_{\mathcal{E},p}(u) : \lambda\psi_{\mathcal{E},p}^2(u) \right)$$

where λ is determined by the ladder algorithm.

Comparison

Test algorithm	Asymptotics		Supersingular input			Non-Supersingular input		
	Time ($\mathbb{F}_p$-ops)	Space ($\mathbb{F}_p$-elts)	MCycles: Avg.	Med.	Stack (B)	MCycles: Avg.	Med.	Stack (B)
Random point	$O(n \log p)$	$O(1)$	63.4	62.2	2890	65.3	62.9	2890
Product tree	$O((\log n)(\log p))$	$O(\log n)$	6.7	6.1	4344	1.7	1.6	3896
Sutherland	$O(\log^2 p)$	$O(1)$	35.4	35.1	**2696**	**0.8**	**0.4**	**2696**
Doliskani	$O(\log p)$	$O(1)$	**4.5**	**4.7**	3280	2.9	2.8	3264

- Using an Intel i7-10610U processor running at 4.90 GHz (see paper for details);
- Cycles were measured using the `bench` utility provided in the CSIDH code package;
- The implementation of the Product tree algorithm was taken directly from the CTIDH library.
- timings: CSIDH-512 parameters using the CSIDH-512 $\mathbb{F}_p$ code

Comparison

Test algorithm	Asymptotics		Supersingular input			Non-Supersingular input		
	Time ($\mathbb{F}_p$-ops)	Space ($\mathbb{F}_p$-elts)	MCycles: Avg.	Med.	Stack (B)	MCycles: Avg.	Med.	Stack (B)
Random point	$O(n \log p)$	$O(1)$	63.4	62.2	2890	65.3	62.9	2890
Product tree	$O((\log n)(\log p))$	$O(\log n)$	6.7	6.1	4344	1.7	1.6	3896
Sutherland	$O(\log^2 p)$	$O(1)$	35.4	35.1	**2696**	**0.8**	**0.4**	**2696**
Doliskani	$O(\log p)$	$O(1)$	**4.5**	**4.7**	3280	2.9	2.8	3264

- Using an Intel i7-10610U processor running at 4.90 GHz (see paper for details);
- Cycles were measured using the `bench` utility provided in the CSIDH code package;
- The implementation of the Product tree algorithm was taken directly from the CTIDH library.
- timings: CSIDH-512 parameters using the CSIDH-512 $\mathbb{F}_p$ code

Conclusion: we suggest using Doliskani for CSIDH key validation.
See `https://ia.cr/2022/880` for details.