Towards practical key exchange from ordinary isogeny graphs

Luca De Feo [1,3] Jean Kieffer [2,3,4] Benjamin Smith [3]

[1]UVSQ, Université Paris Saclay

[2]École normale supérieure, Paris

[3]Inria and École polytechnique, Université Paris Saclay

[4]Inria and IMB, Université de Bordeaux

December 6, 2018

Isogeny-based protocols

Post-quantum candidates for key echange/encapsulation: e.g. SIDH/SIKE.

Inspired by earlier ideas of Couveignes and Rostovtsev–Stolbunov: *CRS key exchange construction*.

Isogeny-based protocols

Post-quantum candidates for key echange/encapsulation: e.g. SIDH/SIKE.

Inspired by earlier ideas of Couveignes and Rostovtsev–Stolbunov: *CRS key exchange construction*.

CRS characteristics w.r.t. SIDH

Pros	Cons

Isogeny-based protocols

Post-quantum candidates for key echange/encapsulation: e.g. SIDH/SIKE.

Inspired by earlier ideas of Couveignes and Rostovtsev–Stolbunov: *CRS key exchange construction*.

CRS characteristics w.r.t. SIDH

Pros	Cons
	▸ Very slow (minutes) ▸ Subexponential quantum attack

Isogeny-based protocols

Post-quantum candidates for key echange/encapsulation: e.g. SIDH/SIKE.

Inspired by earlier ideas of Couveignes and Rostovtsev–Stolbunov: *CRS key exchange construction*.

CRS characteristics w.r.t. SIDH

Pros

- Efficient key validation: post-quantum NIKE
- More "natural" security hypotheses

Cons

- Very slow (minutes)
- Subexponential quantum attack

Isogeny-based protocols

Post-quantum candidates for key echange/encapsulation: e.g. SIDH/SIKE.

Inspired by earlier ideas of Couveignes and Rostovtsev–Stolbunov: *CRS key exchange construction*.

CRS characteristics w.r.t. SIDH

Pros

- Efficient key validation: post-quantum NIKE
- More "natural" security hypotheses

Cons

- Very slow (minutes)
- Subexponential quantum attack

Both: small keys.

Goals

CRS is worth improving.

- Key validation
- Security analysis
- Pre- and post-quantum parameter proposals
- Algorithmic improvements.

Cryptography with a group action

Hard Homogeneous Space (Couveignes): (G, X) where

- G finite commutative group
- $G \circlearrowright X$
- $g \mapsto g \cdot x_0$ is a 1-to-1 correspondence between G and X.

Hardness hypotheses:

- Given g and x, computing $g \cdot x$ is easy
- Given x and $g \cdot x$, computing g is hard.

Cryptography with a group action

Hard Homogeneous Space (Couveignes): (G, X) where

- G finite commutative group
- $G \circlearrowright X$
- $g \mapsto g \cdot x_0$ is a 1-to-1 correspondence between G and X.

Hardness hypotheses:

- Given g and x, computing $g \cdot x$ is easy
- Given x and $g \cdot x$, computing g is hard.

Alice $\qquad\qquad$ x_0 $\qquad\qquad$ Bob

Cryptography with a group action

Hard Homogeneous Space (Couveignes): (G, X) where

- G finite commutative group
- $G \circlearrowright X$
- $g \mapsto g \cdot x_0$ is a 1-to-1 correspondence between G and X.

Hardness hypotheses:

- Given g and x, computing $g \cdot x$ is easy
- Given x and $g \cdot x$, computing g is hard.

Alice		Bob
	x_0	
$a \leftarrow^R G$		$b \leftarrow^R G$

Cryptography with a group action

Hard Homogeneous Space (Couveignes): (G, X) where

- G finite commutative group
- $G \circlearrowright X$
- $g \mapsto g \cdot x_0$ is a 1-to-1 correspondence between G and X.

Hardness hypotheses:

- Given g and x, computing $g \cdot x$ is easy
- Given x and $g \cdot x$, computing g is hard.

Alice		Bob
$a \xleftarrow{R} G$		$b \xleftarrow{R} G$
$x_a \leftarrow a \cdot x_0$		$x_b \leftarrow b \cdot x_0$

$$\begin{tikzcd} & x_0 \arrow[dl, dotted, "a"'] \arrow[dr, dotted, "b"] & \\ x_a & & x_b \end{tikzcd}$$

Cryptography with a group action

Hard Homogeneous Space (Couveignes): (G, X) where

- G finite commutative group
- $G \circlearrowright X$
- $g \mapsto g \cdot x_0$ is a 1-to-1 correspondence between G and X.

Hardness hypotheses:

- Given g and x, computing $g \cdot x$ is easy
- Given x and $g \cdot x$, computing g is hard.

Alice

$$a \xleftarrow{R} G$$
$$x_a \leftarrow a \cdot x_0$$
$$s \leftarrow a \cdot x_b$$

$$\begin{tikzcd}
 & x_0 \arrow[dl, dotted, "a"'] \arrow[dr, dotted, "b"] & \\
x_a \arrow[dr, dotted, "b"'] & & x_b \arrow[dl, dotted, "a"] \\
 & s &
\end{tikzcd}$$

Bob

$$b \xleftarrow{R} G$$
$$x_b \leftarrow b \cdot x_0$$
$$s \leftarrow b \cdot x_a$$

Cryptography with a group action (2)

Hardness hypotheses:

- Given g and x, computing $g \cdot x$ is easy

Cryptography with a group action (2)

Hardness hypotheses:

- Given g and x, if $g \in S$, computing $g \cdot x$ is easy

where S is a small set of generators.

Cryptography with a group action (2)

Hardness hypotheses:

- Given g and x, if $g \in S$, computing $g \cdot x$ is easy

where S is a small set of generators.

The same DH key exchange works:

- Sample $a \leftarrow G$ directly as a product $\prod s_i^{k_i}$, $s_i \in S$
- Compute $a \cdot x$ as the *sequence* of actions of s_i.

The Cayley graph

Computing the group action = walking in the *Cayley graph*:

- $V = X$
- Edge labelled by $s \in S$ between x and $s \cdot x$.

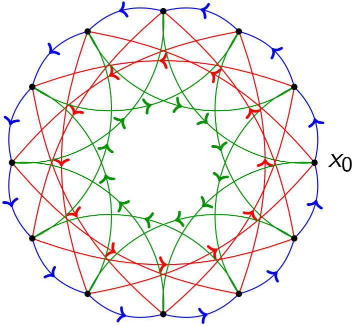
If $S = \{s_1, s_2, s_3\} \cup \{s_1^{-1}, s_2^{-1}, s_3^{-1}\}$:

The Cayley graph

Computing the group action = walking in the *Cayley graph*:

- $V = X$
- Edge labelled by $s \in S$ between x and $s \cdot x$.

If $S = \{s_1, s_2, s_3\} \cup \{s_1^{-1}, s_2^{-1}, s_3^{-1}\}$:



The Cayley graph

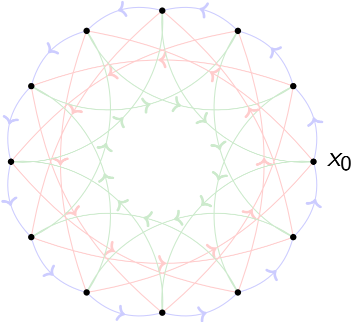
Computing the group action = walking in the *Cayley graph*:

- $V = X$
- Edge labelled by $s \in S$ between x and $s \cdot x$.

If $S = \{s_1, s_2, s_3\} \cup \{s_1^{-1}, s_2^{-1}, s_3^{-1}\}$:

Alice

$a = {s_1}^2 {s_2}^1 {s_3}^{-1}$



The Cayley graph

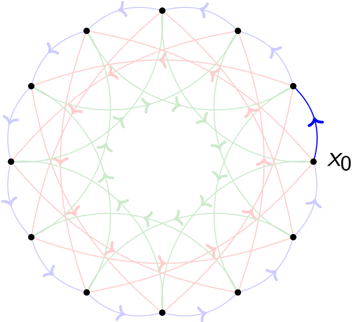
Computing the group action = walking in the *Cayley graph*:

- $V = X$
- Edge labelled by $s \in S$ between x and $s \cdot x$.

If $S = \{s_1, s_2, s_3\} \cup \{s_1^{-1}, s_2^{-1}, s_3^{-1}\}$:

Alice

$a = {s_1}^2 {s_2}^1 {s_3}^{-1}$



The Cayley graph

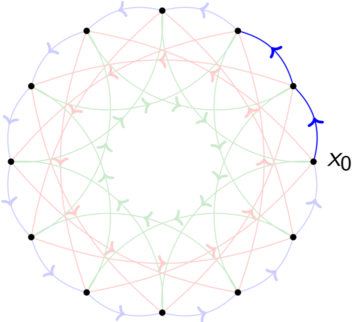
Computing the group action = walking in the *Cayley graph*:

- $V = X$
- Edge labelled by $s \in S$ between x and $s \cdot x$.

If $S = \{s_1, s_2, s_3\} \cup \{s_1^{-1}, s_2^{-1}, s_3^{-1}\}$:

Alice

$a = {s_1}^2 {s_2}^1 {s_3}^{-1}$



The Cayley graph

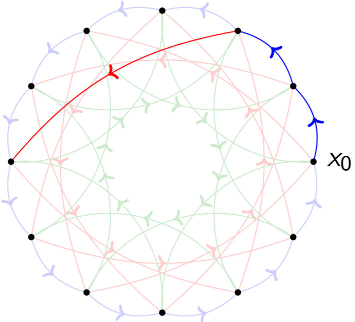
Computing the group action = walking in the *Cayley graph*:

- $V = X$
- Edge labelled by $s \in S$ between x and $s \cdot x$.

If $S = \{s_1, s_2, s_3\} \cup \{s_1^{-1}, s_2^{-1}, s_3^{-1}\}$:

Alice

$a = s_1{}^2 s_2{}^1 s_3{}^{-1}$



The Cayley graph

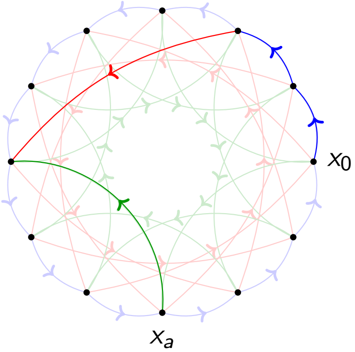
Computing the group action = walking in the *Cayley graph*:

- $V = X$
- Edge labelled by $s \in S$ between x and $s \cdot x$.

If $S = \{s_1, s_2, s_3\} \cup \{s_1^{-1}, s_2^{-1}, s_3^{-1}\}$:

Alice

$a = s_1{}^2 s_2{}^1 s_3{}^{-1}$



The Cayley graph

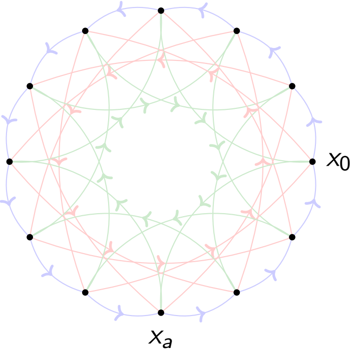
Computing the group action = walking in the *Cayley graph*:

- $V = X$
- Edge labelled by $s \in S$ between x and $s \cdot x$.

If $S = \{s_1, s_2, s_3\} \cup \{s_1^{-1}, s_2^{-1}, s_3^{-1}\}$:

Alice

$a = {s_1}^2 {s_2}^1 {s_3}^{-1}$



Bob

$b = {s_1}^{-2} {s_2}^0 {s_3}^1$

The Cayley graph

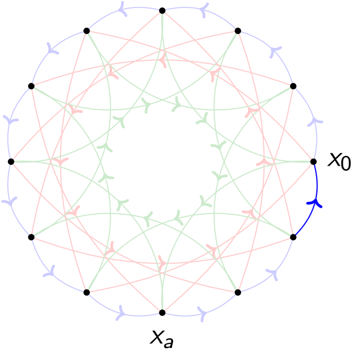
Computing the group action = walking in the *Cayley graph*:

- $V = X$
- Edge labelled by $s \in S$ between x and $s \cdot x$.

If $S = \{s_1, s_2, s_3\} \cup \{s_1^{-1}, s_2^{-1}, s_3^{-1}\}$:

Alice

$a = s_1{}^2 s_2{}^1 s_3{}^{-1}$



Bob

$b = s_1{}^{-2} s_2{}^0 s_3{}^1$

The Cayley graph

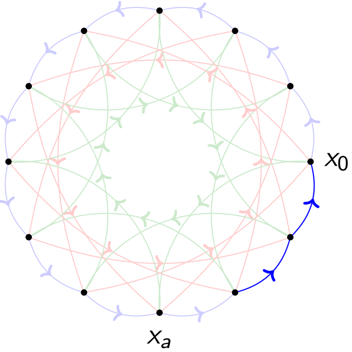
Computing the group action = walking in the *Cayley graph*:

- $V = X$
- Edge labelled by $s \in S$ between x and $s \cdot x$.

If $S = \{s_1, s_2, s_3\} \cup \{s_1^{-1}, s_2^{-1}, s_3^{-1}\}$:

Alice

$a = {s_1}^2 {s_2}^1 {s_3}^{-1}$



Bob

$b = {s_1}^{-2} {s_2}^0 {s_3}^1$

The Cayley graph

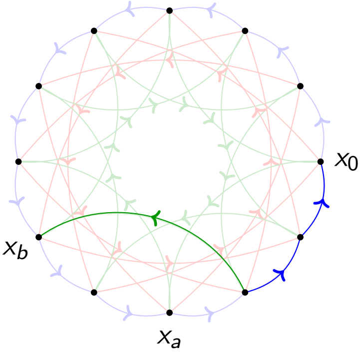
Computing the group action = walking in the *Cayley graph*:

- $V = X$
- Edge labelled by $s \in S$ between x and $s \cdot x$.

If $S = \{s_1, s_2, s_3\} \cup \{s_1^{-1}, s_2^{-1}, s_3^{-1}\}$:

Alice

$a = {s_1}^2 {s_2}^1 {s_3}^{-1}$



Bob

$b = {s_1}^{-2} {s_2}^0 {s_3}^1$

The Cayley graph

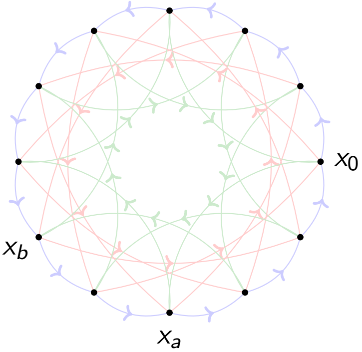
Computing the group action = walking in the *Cayley graph*:

- $V = X$
- Edge labelled by $s \in S$ between x and $s \cdot x$.

If $S = \{s_1, s_2, s_3\} \cup \{s_1^{-1}, s_2^{-1}, s_3^{-1}\}$:

Alice

$a = s_1{}^2 s_2{}^1 s_3{}^{-1}$



Bob

$b = s_1{}^{-2} s_2{}^0 s_3{}^1$

The Cayley graph

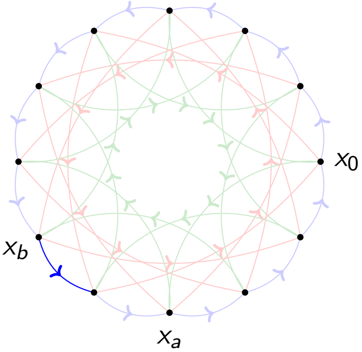
Computing the group action = walking in the *Cayley graph*:

- $V = X$
- Edge labelled by $s \in S$ between x and $s \cdot x$.

If $S = \{s_1, s_2, s_3\} \cup \{s_1^{-1}, s_2^{-1}, s_3^{-1}\}$:

Alice

$a = {s_1}^2 {s_2}^1 {s_3}^{-1}$



Bob

$b = {s_1}^{-2} {s_2}^0 {s_3}^1$

The Cayley graph

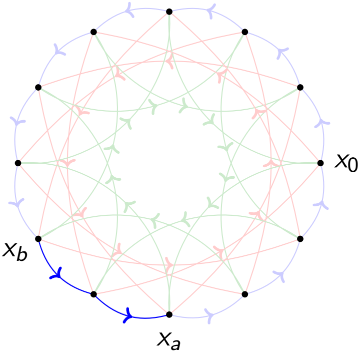
Computing the group action = walking in the *Cayley graph*:

- $V = X$
- Edge labelled by $s \in S$ between x and $s \cdot x$.

If $S = \{s_1, s_2, s_3\} \cup \{s_1^{-1}, s_2^{-1}, s_3^{-1}\}$:

Alice

$a = {s_1}^2 {s_2}^1 {s_3}^{-1}$



Bob

$b = {s_1}^{-2} {s_2}^0 {s_3}^1$

The Cayley graph

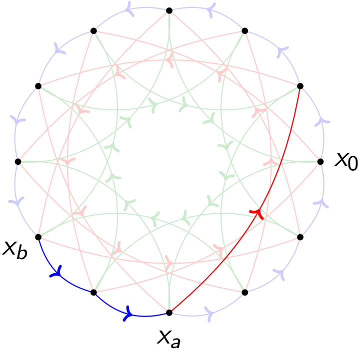
Computing the group action = walking in the *Cayley graph*:

- $V = X$
- Edge labelled by $s \in S$ between x and $s \cdot x$.

If $S = \{s_1, s_2, s_3\} \cup \{s_1^{-1}, s_2^{-1}, s_3^{-1}\}$:

Alice

$a = s_1{}^2 s_2{}^1 s_3{}^{-1}$



Bob

$b = s_1{}^{-2} s_2{}^0 s_3{}^1$

The Cayley graph

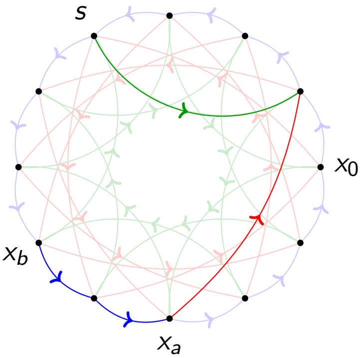
Computing the group action = walking in the *Cayley graph*:

- $V = X$
- Edge labelled by $s \in S$ between x and $s \cdot x$.

If $S = \{s_1, s_2, s_3\} \cup \{s_1^{-1}, s_2^{-1}, s_3^{-1}\}$:

Alice

$a = {s_1}^2 {s_2}^1 {s_3}^{-1}$



Bob

$b = {s_1}^{-2} {s_2}^0 {s_3}^1$

The Cayley graph

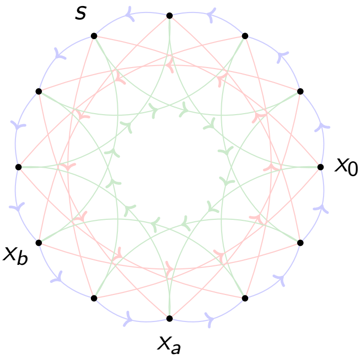
Computing the group action = walking in the *Cayley graph*:

- $V = X$
- Edge labelled by $s \in S$ between x and $s \cdot x$.

If $S = \{s_1, s_2, s_3\} \cup \{s_1^{-1}, s_2^{-1}, s_3^{-1}\}$:

Alice

$a = {s_1}^2 {s_2}^1 {s_3}^{-1}$



Bob

$b = {s_1}^{-2} {s_2}^0 {s_3}^1$

The Cayley graph

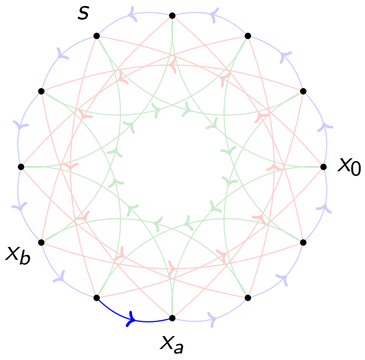
Computing the group action = walking in the *Cayley graph*:

- $V = X$
- Edge labelled by $s \in S$ between x and $s \cdot x$.

If $S = \{s_1, s_2, s_3\} \cup \{s_1^{-1}, s_2^{-1}, s_3^{-1}\}$:

Alice

$a = {s_1}^2 {s_2}^1 {s_3}^{-1}$



Bob

$b = {s_1}^{-2} {s_2}^0 {s_3}^1$

The Cayley graph

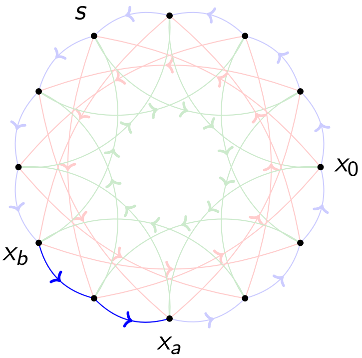
Computing the group action = walking in the *Cayley graph*:

- $V = X$
- Edge labelled by $s \in S$ between x and $s \cdot x$.

If $S = \{s_1, s_2, s_3\} \cup \{s_1^{-1}, s_2^{-1}, s_3^{-1}\}$:

Alice

$a = s_1{}^2 s_2{}^1 s_3{}^{-1}$



Bob

$b = s_1{}^{-2} s_2{}^0 s_3{}^1$

The Cayley graph

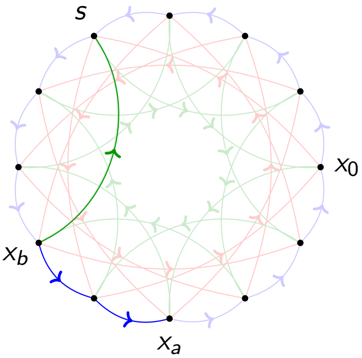
Computing the group action = walking in the *Cayley graph*:

- $V = X$
- Edge labelled by $s \in S$ between x and $s \cdot x$.

If $S = \{s_1, s_2, s_3\} \cup \{s_1^{-1}, s_2^{-1}, s_3^{-1}\}$:

Alice

$a = {s_1}^2 {s_2}^1 {s_3}^{-1}$



Bob

$b = {s_1}^{-2} {s_2}^0 {s_3}^1$

Which HHS could we use?

Where can we find such a (potentially quantum-resistant) Hard Homogeneous Space?

Which HHS could we use?

Where can we find such a (potentially quantum-resistant) Hard Homogeneous Space?

Use isogenies between ordinary elliptic curves:

- X is a set of ordinary elliptic curves
- G is an arithmetic group: *class group*
- S is a set of "small" elements in G
- Computing $s \cdot E$ means computing an *isogeny*.

Why ordinary? Supersingular and ordinary isogeny graphs do not have the same structure.

Elliptic curves and isogenies

- $\mathbb{F}_q$ finite field of large char. p and size q
- E *ordinary* elliptic curve ($\neq$ supersingular) over $\mathbb{F}_q$
- ℓ small prime.

Elliptic curves and isogenies

- $\mathbb{F}_q$ finite field of large char. p and size q
- E *ordinary* elliptic curve ($\neq$ supersingular) over $\mathbb{F}_q$
- ℓ small prime.

ℓ-isogeny

Algebraic morphism ϕ between two elliptic curves, of *degree* ℓ:

- Given by rational fractions of degree ℓ
- ℓ-to-1, in particular $\#\operatorname{Ker}\phi = \ell$.

Elliptic curves and isogenies

- $\mathbb{F}_q$ finite field of large char. p and size q
- E *ordinary* elliptic curve ($\neq$ supersingular) over $\mathbb{F}_q$
- ℓ small prime.

ℓ-isogeny

Algebraic morphism ϕ between two elliptic curves, of *degree* ℓ:

- Given by rational fractions of degree ℓ
- ℓ-to-1, in particular $\#\operatorname{Ker}\phi = \ell$.

Endomorphism = isogeny $E \to E$ (or 0).
Commutative endomorphism ring $\operatorname{End}(E)$.

Elliptic curves and isogenies

- $\mathbb{F}_q$ finite field of large char. p and size q
- E *ordinary* elliptic curve ($\neq$ supersingular) over $\mathbb{F}_q$
- ℓ small prime.

ℓ-isogeny

Algebraic morphism ϕ between two elliptic curves, of *degree* ℓ:

- Given by rational fractions of degree ℓ
- ℓ-to-1, in particular $\#\operatorname{Ker}\phi = \ell$.

Endomorphism = isogeny $E \to E$ (or 0).
Commutative endomorphism ring $\operatorname{End}(E)$.

Fix $\mathcal{O}$ and take $X = \{E \text{ ordinary ell. curve} \mid \operatorname{End}(E) = \mathcal{O}\}$.

Isogenies/ideals correspondence

$E \in X$, i.e. $\mathrm{End}(E) = \mathcal{O}$.

Isogenies from E		Ideals in $\mathcal{O}$
ℓ-isogeny $\phi : E \to E'$	$\longleftrightarrow$	Ideal $\mathfrak{l}$ of norm ℓ in $\mathcal{O}$ $= \{\beta$ vanishing on $\mathrm{Ker}\,\phi\}$
Endomorphism $\alpha : E \to E$	$\longleftrightarrow$	Principal ideal (α)

Isogenies/ideals correspondence

$E \in X$, i.e. $\mathrm{End}(E) = \mathcal{O}$.

Isogenies from E		Ideals in $\mathcal{O}$
ℓ-isogeny $\phi: E \to E'$	$\longleftrightarrow$	Ideal $\mathfrak{l}$ of norm ℓ in $\mathcal{O}$ $= \{\beta$ vanishing on $\mathrm{Ker}\,\phi\}$
Endomorphism $\alpha: E \to E$	$\longleftrightarrow$	Principal ideal (α)

Group action (*complex multiplication*)

Define $\mathfrak{l} \cdot E = E'$: codomain of the corresponding ℓ-isogeny.

Isogenies/ideals correspondence

$E \in X$, i.e. $\mathrm{End}(E) = \mathcal{O}$.

Isogenies from E		Ideals in $\mathcal{O}$
ℓ-isogeny $\phi: E \to E'$	$\longleftrightarrow$	Ideal $\mathfrak{l}$ of norm ℓ in $\mathcal{O}$ $= \{\beta$ vanishing on $\mathrm{Ker}\,\phi\}$
Endomorphism $\alpha: E \to E$	$\longleftrightarrow$	Principal ideal (α)

Group action (*complex multiplication*)

Define $\mathfrak{l} \cdot E = E'$: codomain of the corresponding ℓ-isogeny.

- G is the *class group* of $\mathcal{O}$: ideals modulo principal ideals.
- S is a set of ideals with small prime norms ℓ_i.
  When ℓ_i is nice (*split*), two ideals of norm ℓ_i: $\mathfrak{l}_i$ and $\mathfrak{l}_i^{-1}$.

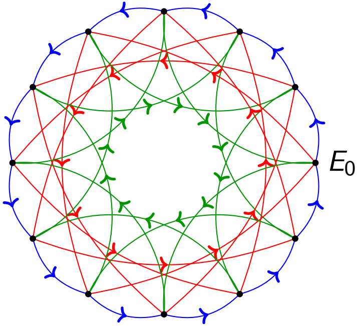
Group action of G on X, which we use as a HHS.

Isogeny walks

Computing the group action = walking in the *isogeny graph*:

- Vertices are elliptic curves,
- Edges are isogenies labelled per degree ℓ_i (arrows give the action of $\mathfrak{l}_i$).

$a = (2, 1, -1)$ represents the ideal $\mathfrak{a} = \mathfrak{l}_1{}^2 \mathfrak{l}_2{}^1 \mathfrak{l}_3{}^{-1}$:

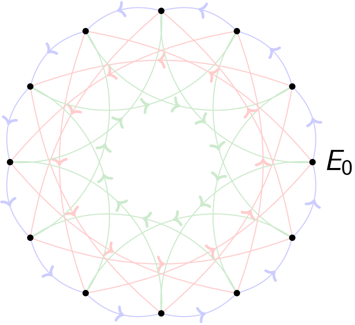


Isogeny walks

Computing the group action = walking in the *isogeny graph*:

- Vertices are elliptic curves,
- Edges are isogenies labelled per degree ℓ_i (arrows give the action of $\mathfrak{l}_i$).

$a = (2, 1, -1)$ represents the ideal $\mathfrak{a} = {\mathfrak{l}_1}^2 {\mathfrak{l}_2}^1 {\mathfrak{l}_3}^{-1}$:

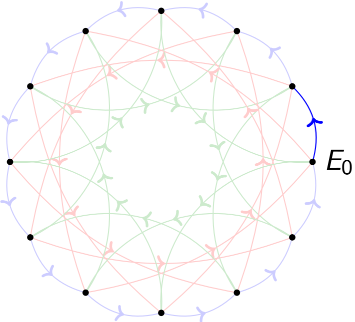


Isogeny walks

Computing the group action = walking in the *isogeny graph*:

- Vertices are elliptic curves,
- Edges are isogenies labelled per degree ℓ_i (arrows give the action of $\mathfrak{l}_i$).

$a = (2, 1, -1)$ represents the ideal $\mathfrak{a} = \mathfrak{l}_1{}^2 \mathfrak{l}_2{}^1 \mathfrak{l}_3{}^{-1}$:

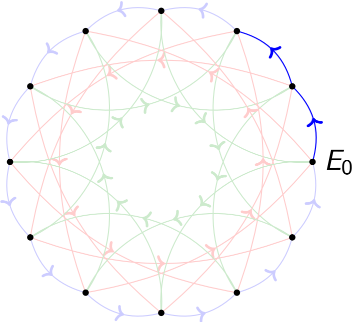


Isogeny walks

Computing the group action = walking in the *isogeny graph*:

- Vertices are elliptic curves,
- Edges are isogenies labelled per degree ℓ_i (arrows give the action of $\mathfrak{l}_i$).

$a = (2, 1, -1)$ represents the ideal $\mathfrak{a} = \mathfrak{l}_1{}^2 \mathfrak{l}_2{}^1 \mathfrak{l}_3{}^{-1}$:

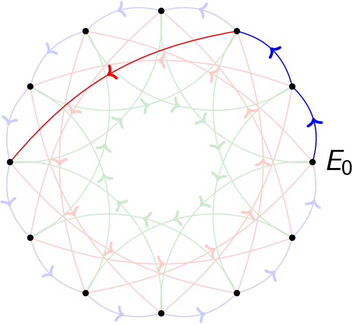


Isogeny walks

Computing the group action = walking in the *isogeny graph*:

- Vertices are elliptic curves,
- Edges are isogenies labelled per degree ℓ_i (arrows give the action of $\mathfrak{l}_i$).

$a = (2, 1, -1)$ represents the ideal $\mathfrak{a} = \mathfrak{l}_1{}^2 \mathfrak{l}_2{}^1 \mathfrak{l}_3{}^{-1}$:

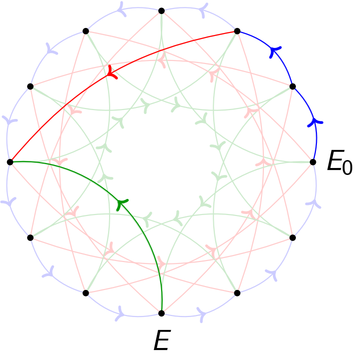


Isogeny walks

Computing the group action = walking in the *isogeny graph*:

- Vertices are elliptic curves,
- Edges are isogenies labelled per degree ℓ_i (arrows give the action of $\mathfrak{l}_i$).

$a = (2, 1, -1)$ represents the ideal $\mathfrak{a} = \mathfrak{l}_1{}^2 \mathfrak{l}_2{}^1 \mathfrak{l}_3{}^{-1}$: $\quad E = \mathfrak{a} \cdot E_0$.



Key validation

E is valid protocol data iff $\mathrm{End}(E) = \mathcal{O}$.

This can be checked using

- a few scalar multiplications on E,
- a few small-degree isogenies.

Key validation is easy and efficient.

Hardness assumptions

Isogeny DH-analogues:

- Class Group Action-DDH (CGA-DDH)
- CGA-CDH

Sampling in G using products of small ideals is a probability distribution σ.

- Distinguish σ from the uniform distribution: Isogeny Walk Distinguishing (IWD).

Security analysis

Theorem (assuming GRH, IWD, CGA-DDH)

The key exchange protocol is session-key secure in the authenticated-links adversarial model of Canetti–Krawczyk.

Theorem (assuming IWD, CGA-CDH)

The derived hashed ElGamal protocol is IND-CPA secure in the random oracle model.

Key validation gives CCA-secure encryption. In contrast, CCA attack against SIKE.PKE (Galbraith et al., AsiaCrypt 2016).

Classical security

CGA-DDH

Compute an isogeny between two curves to recover the key.
Best classical algorithm: $O(\sqrt{N})$ where $N = \#G \simeq \sqrt{q}$.

- Choose $\log_2(q) \simeq 4n$.

IWD

Heuristic: it is enough to have keyspace size $\geq \sqrt{q}$.
We cannot prove this even under GRH.

- Keyspace size: isogeny degrees $\ell_i = O(\log q)$.

Quantum security

Key recovery is an instance of the Hidden Shift Problem.

- Kuperberg's algorithm solves HShP in subexponential time.

Quantum security

Key recovery is an instance of the Hidden Shift Problem.

- Kuperberg's algorithm solves HShP in subexponential time.
- This does not mean that CRS is broken.
- Estimates on query complexity alone:
  $\log_2(q) = 688,\ 1656,\ 3068$ for NIST levels 1, 3, 5.

Computing small-degree isogenies

The basic building block of CRS is computing ℓ-isogenies.

Computing small-degree isogenies

The basic building block of CRS is computing ℓ-isogenies.

The CRS approach

Use *modular equations* linking E and E'.

- Find the roots of a degree $\ell + 1$ polynomial over $\mathbb{F}_q$.

Computing small-degree isogenies

The basic building block of CRS is computing ℓ-isogenies.

The CRS approach

Use *modular equations* linking E and E'.

- Find the roots of a degree $\ell + 1$ polynomial over $\mathbb{F}_q$.

Our contribution

Suppose there is some $P \in E(\mathbb{F}_q)$ of order ℓ.

- Find one such P using a scalar multiplication on E,
- Compute the image curve knowing the kernel $\langle P \rangle$.

Computing small-degree isogenies

The basic building block of CRS is computing ℓ-isogenies.

The CRS approach

Use *modular equations* linking E and E'.

- Find the roots of a degree $\ell + 1$ polynomial over $\mathbb{F}_q$.

Our contribution

Suppose there is some $P \in E(\mathbb{F}_q)$ of order ℓ.

- Find one such P using a scalar multiplication on E,
- Compute the image curve knowing the kernel $\langle P \rangle$.

Cost analysis

ℓ-torsion point	Modular equation

Computing small-degree isogenies

The basic building block of CRS is computing ℓ-isogenies.

The CRS approach

Use *modular equations* linking E and E'.

- Find the roots of a degree $\ell + 1$ polynomial over $\mathbb{F}_q$.

Our contribution

Suppose there is some $P \in E(\mathbb{F}_q)$ of order ℓ.

- Find one such P using a scalar multiplication on E,
- Compute the image curve knowing the kernel $\langle P \rangle$.

Cost analysis

ℓ-torsion point	Modular equation
$O(\log(q) + \ell)$	

Computing small-degree isogenies

The basic building block of CRS is computing ℓ-isogenies.

The CRS approach

Use *modular equations* linking E and E'.

- Find the roots of a degree $\ell + 1$ polynomial over $\mathbb{F}_q$.

Our contribution

Suppose there is some $P \in E(\mathbb{F}_q)$ of order ℓ.

- Find one such P using a scalar multiplication on E,
- Compute the image curve knowing the kernel $\langle P \rangle$.

Cost analysis

ℓ-torsion point	Modular equation
$O(\log(q) + \ell)$	$O(\ell^2 \log q)$

Computing small-degree isogenies

The basic building block of CRS is computing ℓ-isogenies.

The CRS approach

Use *modular equations* linking E and E'.

- Find the roots of a degree $\ell+1$ polynomial over $\mathbb{F}_q$.

Our contribution

Suppose there is some $P \in E(\mathbb{F}_q)$ of order ℓ.

- Find one such P using a scalar multiplication on E,
- Compute the image curve knowing the kernel $\langle P \rangle$.

Cost analysis

ℓ-torsion point		Modular equation
$O(\log(q) + \ell)$	$\ll$	$O(\ell^2 \log q)$

The twisting trick

Suppose $P \in E$ of order ℓ_i allows to compute the action of $\mathfrak{l}_i$. Can we also compute efficiently the action of $\mathfrak{l}_i^{-1}$?

The twisting trick

Suppose $P \in E$ of order ℓ_i allows to compute the action of $\mathfrak{l}_i$. Can we also compute efficiently the action of $\mathfrak{l}_i^{-1}$?

The twisting trick

Suppose $q = -1 \mod \ell_i$. Then E^t (quad. twist) also has a point of order ℓ_i.

- We can efficiently compute the action of $\mathfrak{l}_i^{-1}$ by twisting back and forth.

The twisting trick

Suppose $P \in E$ of order ℓ_i allows to compute the action of $\mathfrak{l}_i$. Can we also compute efficiently the action of $\mathfrak{l}_i^{-1}$?

The twisting trick

Suppose $q = -1 \mod \ell_i$. Then E^t (quad. twist) also has a point of order ℓ_i.

- We can efficiently compute the action of $\mathfrak{l}_i^{-1}$ by twisting back and forth.

Why? The Frobenius on $E[\ell_i]$ is $\begin{pmatrix} 1 & 0 \\ 0 & q \end{pmatrix}$, so the Frobenius on $E^t[\ell_i]$ is $\begin{pmatrix} -1 & 0 \\ 0 & -q \end{pmatrix}$ and $-q = 1$.

Finding good initial curves

More small-order points on E_0 = more efficient cryptosystem.

Finding good initial curves

More small-order points on E_0 = more efficient cryptosystem.

Only exponential algorithms are known to find ordinary curves with smooth order (no CM method here).

We look for E_0 using

- early-abort point counting
- curve selection with modular curves

but we cannot use our improvements in full even after 2 years CPU time searching.

Best results

512-bit prime $q = 7\prod \ell_i - 1$, where the ℓ_i are all primes ≤ 380.

Best E_0:

$$\#E_0(\mathbb{F}_q) = 3 \cdot 5 \cdot 7 \cdot 11 \cdot 13 \cdot 17 \cdot 103 \cdot 523 \cdot 821 \cdot R$$
$$\#E_0^t(\mathbb{F}_q) = (\text{same } \leq 103) \cdot 947 \cdot 1723 \cdot R'$$

Discriminant $\Delta = -2^3 \cdot$ squarefree.

Best results

512-bit prime $q = 7 \prod \ell_i - 1$, where the ℓ_i are all primes ≤ 380.

Best E_0:

$$\#E_0(\mathbb{F}_q) = 3 \cdot 5 \cdot 7 \cdot 11 \cdot 13 \cdot 17 \cdot 103 \cdot 523 \cdot 821 \cdot R$$
$$\#E_0^t(\mathbb{F}_q) = (\text{same } \leq 103) \cdot 947 \cdot 1723 \cdot R'$$

Discriminant $\Delta = -2^3 \cdot$ squarefree.

Type	Isogeny degrees	#steps
Torsion ($\mathbb{F}_q$)	11: see above	409
Torsion ($\mathbb{F}_{q^r}$)	13: $19, 661$ ($r = 3$), ...	81 down to 10
General	25: $73, 89, \ldots$ up to 359	6 down to 1

Not enough primes in the first two lines: walk $\simeq 520$ s.

Take away messages

- Isogeny graphs can be used to construct post-quantum key exchange protocols, and post-quantum NIKE.
- Our improvements speed up CRS considerably, but we cannot use them in full with ordinary curves (not enough torsion points!)
  See next talk on CSIDH.