Advances in isogeny-based cryptography

Benjamin Smith

Arithmetic of low-dimensional abelian varieties // ICERM // 6/6/19

Inria + Laboratoire d'Informatique de l'École polytechnique (LIX)

Elliptic curve cryptography

Classic Diffie–Hellman key exchange in a group $\mathcal{G} = \langle P \rangle \cong \mathbb{Z}/N\mathbb{Z}$

Phase 1 **Alice** samples a secret $a \in \mathbb{Z}/N\mathbb{Z}$;
computes $A := [a]P$ and publishes A
Bob samples a secret $b \in \mathbb{Z}/N\mathbb{Z}$;
computes $B := [b]P$ and publishes B

Breaking keypairs (e.g. recovering a from A) = **Discrete Logarithm Problem (DLP)**.

Phase 2 **Alice** computes $S = [a]B$.
Bob computes $S = [b]A$.

The protocol correctly computes a **shared secret** because

$$A = [a]P \qquad B = [b]P \qquad S = [ab]P$$

Computational Diffie–Hellman Problem (CDHP): recovering S from (P, A, B).

Curve-based cryptography

Elliptic curves are the **gold standard** source of groups for DLP-based crypto.

The **best known algorithm** for solving DLP instances in $\mathcal{E}(\mathbb{F}_p)$ for general prime-order $\mathcal{E}$ is *still* **Pollard** ρ, in $O(\sqrt{p})$ group operations.

The weak curves (pairing-friendly, anomalous, ...) are easy to identify and avoid.

Generalizing from elliptic curves to **higher-dimensional AVs** is obvious:

- dimension g over $\mathbb{F}_q$ give groups of size $\sim q^g$;
- compressed keys encode to $g \log_q$ bits;
- efficient representation and arithmetic is tricky (but let's be optimistic...)
- constructing secure instances is a nightmare (but let's be really optimistic...)

The **bottom line**: for g-dimensional AVs to be competitive with elliptic curves, we need DLP hardness close to $O(q^{g/2})$.

Attacks in higher dimensions

Unfortunately, **index calculus** algorithms for solving DLPs work better and better as the dimension of the abelian variety grows. We want $\widetilde{O}(q^{g/2})$, but...

- Jacobians of genus-g curves: Gaudry–Thomé–Thériault–Diem in $\widetilde{O}(q^{2-2/g})$
- Jacobians of smooth degree-d plane curves: Diem in $\widetilde{O}(q^{2-2/d})$
- Jacobians of genus-3 hyperelliptic curves: reduce to nonhyperelliptic using isogenies (degenerate Recillas: S. 2007, Frey–Kani 2011) then Diem in $\widetilde{O}(q)$.
- General PPAVs, dim $g > 3$: essentially wiped out by Gaudry in $\widetilde{O}(q^{2-2/g})$.

Result: abelian varieties of dimension ≥ 3 are cryptographically inefficient.

For **constructive cryptographic applications**, we're down to **genus 1 and 2**.

Modern elliptic-curve cryptography

Modern Elliptic Curve Diffie–Hellman (ECDH)

Classic ECDH is just classic DH with $\mathcal{E}(\mathbb{F}_q)$ in place of $\mathbb{G}_m(\mathbb{F}_q)$:

$$A = [a]P \qquad B = [b]P \qquad S = [ab]P$$

Miller (1985) suggested ECDH using **only x-coordinates:**

$$\begin{aligned} A &= x([a]P) & B &= x([b]P) & S &= x([ab]P) \\ &= \pm[a]P & &= \pm[b]P & &= \pm[ab]P \end{aligned}$$

Compute $x(Q) \mapsto x([m]Q)$ with efficient **differential addition chains** such as the **Montgomery ladder**.

Definitive example: `Curve25519` (Bernstein 2006), the benchmark for conventional DH (and now standard in OpenSSH and TLS 1.3).

Even better performance from **Kummer surfaces** with rich 2-torsion structure.

Modern ECDH: where is the group?

x-only ECDH works because **Diffie–Hellman has no explicit group operation**.

$$A = [a]P \qquad B = [b]P \qquad S = [ab]P$$

Formally, we have **an "action" of $\mathbb{Z}$ on a set $\mathcal{X}$** (here, $\mathcal{X} = \mathcal{G}/\langle \pm 1 \rangle$).

In fact, the quotient structure $\mathcal{G}/\langle \pm 1 \rangle$ is crucial: it facilitates

- **security proofs** by relating CDHPs in $\mathcal{X}$ and $\mathcal{G}$
- **efficient evaluation** of the $\mathbb{Z}$-action on $\mathcal{X}$: the group op on $\mathcal{G}$ induces an operation $(\pm P, \pm Q, \pm(P - Q)) \mapsto \pm(P + Q)$ on $\mathcal{X}$, which we use to compute $(m, x(P)) \mapsto x([m]P)$ using differential addition chains.

The quantum menace

Elliptic curve crypto is state-of-the-art.

Genus-2 crypto is an aggressive alternative.

But both are based on the hardness of DLP, which **Shor's quantum algorithm** solves in **polynomial time**.

Attacking real-world DH instances with Shor requires **large, general-purpose quantum computers**. *Q: Will sufficiently large quantum computers ever be built?*

*Say **yes** if you want to get funded.*

Global research effort: replacing classic group-based public-key cryptosystems with **postquantum** alternatives.

Classical isogeny-based crypto

Principal homogeneous spaces

Let $\mathfrak{G}$ be a finite commutative group acting on a set $\mathcal{X}$, so

$$\mathfrak{a} \cdot (\mathfrak{b} \cdot P) = \mathfrak{a}\mathfrak{b} \cdot P \qquad \forall \mathfrak{a}, \mathfrak{b} \in \mathfrak{G}, \quad \forall P \in \mathcal{X} \,.$$

$\mathcal{X}$ is a **principal homogeneous space** (PHS) under $\mathfrak{G}$ if

$$P, Q \in \mathcal{X} \implies \exists!\ \mathfrak{g} \in \mathfrak{G} \text{ such that } Q = \mathfrak{g} \cdot P \,.$$

Example: a vector space $\mathfrak{G}$ acting on its underlying affine space $\mathcal{X}$.

The isogeny PHS

Key example of a PHS from **CM theory** for a quadratic imaginary field K:

Group: $\mathfrak{G} = \mathrm{Cl}(O_K)$, the group of ideal classes of the maximal order of K

Space: $\mathcal{X} = \{\mathcal{E}/\mathbb{F}_q \mid \mathrm{End}(\mathcal{E}) \cong O_K\}/(\mathbb{F}_q\text{-isomorphism})$

Action: Ideals $\mathfrak{a}$ in O_K correspond to **isogenies** $\phi_{\mathfrak{a}} : \mathcal{E} \to \mathcal{E}/\mathcal{E}[\mathfrak{a}] =: \mathfrak{a} \cdot \mathcal{E}$.
This action extends to fractional ideals and factors through $\mathrm{Cl}(O_K)$.

We have $\#\mathfrak{G} = \#\mathcal{X} \sim \sqrt{|\Delta|}$, where $\Delta = \mathrm{disc}(O_K) \sim q$.

Forgotten identities

A PHS is like a copy of $\mathfrak{G}$ with the identity $1_{\mathfrak{G}}$ forgotten.

For each $P \in \mathcal{X}$, the map $\varphi_P : \mathfrak{g} \mapsto \mathfrak{g} \cdot P$ is a bijection $\mathfrak{G} \to \mathcal{X}$.

Each φ_P endows $\mathcal{X}$ with the structure of $\mathfrak{G}$, with P as the identity element, via

$$(\mathfrak{a} \cdot P)(\mathfrak{b} \cdot P) = \varphi_P(\mathfrak{a})\varphi_P(\mathfrak{b}) := \varphi_P(\mathfrak{a}\mathfrak{b}) = (\mathfrak{a}\mathfrak{b}) \cdot P\,.$$

Each choice of P yields a different group law on $\mathcal{X}$.

A Diffie–Hellman analogue

We have an **obvious analogy** between Group-DH and PHS-DH:

$$A = [a]P \qquad B = [b]P \qquad S = [ab]P$$

$$A = \mathfrak{a} \cdot P \qquad B = \mathfrak{b} \cdot P \qquad S = \mathfrak{ab} \cdot P$$

Security: need PHS analogues of DLP and CDHP to be hard.

Utility: need to be able to

- efficiently sample uniformly from a sufficiently large keyspace $K \subset \mathfrak{G}$
- efficiently compute the action $(\mathfrak{a}, P) \mapsto \mathfrak{a} \cdot P$ for $\mathfrak{a} \in K$

For the CM PHS, sampling random $\mathfrak{a} \in \mathrm{Cl}(\mathcal{O}_K)$ is easy, but computing an isogeny with kernel $\mathfrak{a}$ is exponential in $N(\mathfrak{a})$. Couveignes suggested smoothing $\mathfrak{a}$ to an equivalent $\prod_i \mathfrak{l}_i^{e_i}$ (with small prime $\mathfrak{l}_i$) using LLL, then acting by the $\mathfrak{l}_i$ in serial.

Vectorization (VEC: breaking public keys):

Given P and Q in $\mathcal{X}$, compute the (unique) $\mathfrak{g} \in \mathfrak{G}$ s.t. $Q = \mathfrak{g} \cdot P$.

$$\begin{tikzcd} P \arrow[r, dashed, "\mathfrak{g}"] & Q \end{tikzcd}$$

Parallelization (PAR: recovering shared secrets):

Given P, A, B in $\mathcal{X}$ with $A = \mathfrak{a} \cdot P$, $B = \mathfrak{b} \cdot P$, compute $S = (\mathfrak{a}\mathfrak{b}) \cdot P$.

$$\begin{tikzcd} P \arrow[r, dashed, "\mathfrak{a}"] \arrow[dr, dashed, "\mathfrak{b}"'] & A \arrow[dr, dashed, "\mathfrak{b}"] & \\ & B \arrow[r, dashed, "\mathfrak{a}"'] & S \end{tikzcd}$$

Hard homogeneous spaces

A **Hard Homogeneous Space (HHS)** is a PHS where VEC and PAR are computationally infeasible.

- The vector/affine space PHS is **not** an HHS.
- The CM PHS is a **conjectural HHS**.

We have a lot **intuition** and folklore about DLP and CDHP.

- Decades of algorithmic study
- Conditional polynomial-time equivalences

What carries over to VEC and PAR?

How hard are hard homogeneous spaces?

Obviously, if we can solve VECs

$$(P, Q = \mathfrak{x} \cdot P) \longmapsto \mathfrak{x}\,,$$

then we can solve PARs

$$(P, A = \mathfrak{a} \cdot P, B = \mathfrak{b} \cdot P) \longmapsto S = \mathfrak{a}\mathfrak{b} \cdot P\,.$$

Let's focus on VEC for a moment.

We can solve any DLP classically in time $O(\sqrt{N})$
using Pollard's ρ or Shanks' Baby-step giant-step.

We can solve VEC in time $O(\sqrt{N})$ using the same algorithms!

Baby-step giant-step: the same for DLP and Vec

Algorithm 1: BSGS in $\mathfrak{G}$

Input: $\mathfrak{g}$ and $\mathfrak{h}$ in $\mathfrak{G}$

Output: x such that $\mathfrak{h} = \mathfrak{g}^x$

$\beta \leftarrow \lceil \sqrt{\#\mathfrak{G}} \rceil$

$(\mathfrak{s}_i) \leftarrow (\mathfrak{g}^i : 1 \leq i \leq \beta)$

Sort/hash $((\mathfrak{s}_i, i))_{i=1}^{\beta}$

$\mathfrak{t} \leftarrow \mathfrak{h}$

for j ***in*** $(1, \ldots, \beta)$ **do**

  if $\mathfrak{t} = \mathfrak{s}_i$ ***for some*** i **then**

    return $i - j\beta$

  $\mathfrak{t} \leftarrow \mathfrak{g}^{\beta}\mathfrak{t}$

return $\perp$ `// Only if` $\mathfrak{h} \notin \langle \mathfrak{g} \rangle$

Algorithm 2: BSGS in $(\mathfrak{G}, \mathcal{X})$

Input: P and Q in $\mathcal{X}$; a generator $\mathfrak{g}$ for $\mathfrak{G}$

Output: x such that $Q = \mathfrak{g}^x \cdot P$

$\beta \leftarrow \lceil \sqrt{\#\mathfrak{G}} \rceil$

$(P_i) \leftarrow (\mathfrak{g}^i \cdot P : 1 \leq i \leq \beta)$

Sort/hash $((P_i, i))_{i=1}^{\beta}$

$T \leftarrow Q$

for j ***in*** $(1, \ldots, \beta)$ **do**

  if $T = P_i$ ***for some*** i **then**

    return $i - j\beta$

  $T \leftarrow \mathfrak{g}^{\beta} \cdot T$

return $\perp$ `// Only if` $Q \notin \langle \mathfrak{e} \rangle \cdot P$

Classical security of the isogeny PHS

Generic algorithms solve Vec in any PHS $(\mathfrak{G}, \mathcal{X})$ in time $O(\sqrt{\#\mathfrak{G}})$.

In the case of the CM PHS, where $\#\mathfrak{G} = \#\mathrm{Cl}(O_K) \sim \sqrt{q}$, the best classical algorithm to compute unknown isogenies runs in time $O(\sqrt{\#\mathfrak{G}}) = O(q^{1/4})$ (Galbraith–Hess–Smart 2002).

But what about using the structure of $\mathfrak{G}$?

Classical limits of the DLP-Vec analogy: Pohlig–Hellman

The **Pohlig–Hellman** algorithm exploits subgroups of $\mathfrak{G}$ to solve DLP instances in time $\widetilde{O}(\sqrt{\text{largest prime factor of } \#\mathfrak{G}})$.

Simplest case: $\#\mathfrak{G} = \prod_i \ell_i$, with the ℓ_i prime.
To find x such that $\mathfrak{h} = \mathfrak{g}^x$, for each i we

1. compute $\mathfrak{h}_i \leftarrow \mathfrak{h}^{m_i}$ and $\mathfrak{g}_i \leftarrow \mathfrak{g}^{m_i}$, where $m_i = \#\mathfrak{G}/\ell_i$;
2. compute x_i such that $\mathfrak{h}_i = \mathfrak{g}_i^{x_i}$ (DLP in order-ℓ_i subgroup)

We then recover x from the (x_i, ℓ_i) using the CRT.

Problem: the **HHS analogue of Step 1 is supposedly hard**!
(Computing $Q_i = \mathfrak{g}^i \cdot P$ where $Q = \mathfrak{g} \cdot P$ is an instance of Par.)

No Pohlig–Hellman

Funny: We don't know how to exploit the structure of $\mathfrak{G}$ to accelerate VEC *or* PAR.

Surprise: classical acceleration **shouldn't exist** in general. **Why?**

- Choose p from a family of primes s.t. all prime factors of $p-1$ are in $o(p)$.
- Now take a black-box group $\mathcal{G}$ of order p.
- **Shoup's theorem**: DLP($\mathcal{G}$) is in $\Theta(\sqrt{p})$.
- Exponentiation yields a PHS $(\mathfrak{G}, \mathcal{X}) = ((\mathbb{Z}/p\mathbb{Z})^\times, \mathcal{G} \setminus \{0\})$,
  and VEC in $(\mathfrak{G}, \mathcal{X})$ solves DLP in $\mathcal{G}$.
- Now $\#\mathfrak{G} = p - 1$, whose prime factors are in $o(p)$,
  so classical subgroup DLPs and VECs are in $o(\sqrt{p})$;
  a HHS Pohlig–Hellman analogue would **contradict Shoup**.

Postquantum isogenies

How not to publish your work

1997: Couveignes submitted to Crypto; rejected. Forgotten. **2007**: published in an obscure SMF special issue, with an extremely helpful title and abstract.

QUELQUES MATHÉMATIQUES DE LA CRYPTOLOGIE À CLÉS PUBLIQUES

par

Jean-Marc Couveignes

Résumé. — Cette note présente quelques développements mathématiques plus ou moins récents de la cryptologie à clés publiques.

Abstract **(A few mathematical tools for public key cryptology)**
I present examples of mathematical objects that are of interest for public key cryptography.

Rediscovery

2006: **Stolbunov and Rostovtsev** independently rediscovered Couveignes' isogeny-based key exchange.

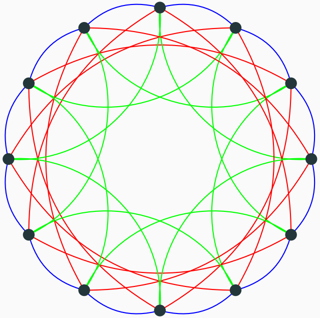
Essential differences:

- Instead of sampling a secret $\mathfrak{a} \in \mathrm{Cl}(\mathcal{O}_K)$ then smoothing it to $\mathfrak{a} \sim \prod_i \mathfrak{l}_i^{e_i}$, they fix a set of small $\mathfrak{l}_i$ and **sample exponent vectors** $(e_1, \ldots, e_n)$, hoping that this is close to uniform on $\mathrm{Cl}(\mathcal{O}_K)$.
- Everything is expressed in terms of walks in **isogeny graphs**, which had come into fashion since Kohel's thesis (1996).
- Rostovtsev and Stolbunov claimed **postquantum security**.

Stolbunov presented this at a 2006 workshop at LIX, which prompted Couveignes to preprint his forgotten seminar notes.

Isogeny graphs

The ℓ-isogeny graphs of curves with CM by O_K are all **cylic**.



A key $\prod_i \mathfrak{l}_i^{e_i}$ corresponds to

1. e_1 steps in the ℓ_1-isogeny graph, then
2. e_2 steps in the ℓ_2-isogeny graph, then
3. e_3 steps in the ℓ_3-isogeny graph,
4. More walks ...

Rostovtsev and Stolbunov's proof-of-concept implementation: **extremely slow**.
We re-implemented it at NIST security level 1: $\geq$ 2000s per DH.

Why is this post-quantum?

We've seen classical generic DLP algorithms solve VEC instances,
so you might think quantum DLP algorithms should solve VEC instances too.

Shor's algorithm solves DLP in polynomial time, but **not** VEC.

VEC is an instance of the abelian **hidden shift problem**: solve using variants of **Kuperberg's algorithm** in quantum **subexponential** time $L_N[1/2, 0]$.
$\implies$ upper bound for **quantum VEC hardness** is $L_N(1/2)$ quantum actions.
$\implies$ upper bound for **quantum PAR hardness** is $L_N(1/2)$ quantum actions.

In a sense, BSGS and Pollard ρ are actually **PHS algorithms**
(with $\mathfrak{G}$ acting on itself), not group algorithms!

Quantum equivalence of VEC and PAR

Galbraith–Panny–S.–Vercauteren (2019):
Unconditional **quantum** polynomial equivalence PAR $\iff$ VEC.

VEC $\implies$ PAR: obvious. PAR $\implies$ VEC:

1. Compute basis $\{\mathfrak{g}_1, \ldots, \mathfrak{g}_r\}$ for $\mathfrak{G}$ using Kitaev/Shor (if not already known)
2. Quantum PAR circuit $(P, \mathfrak{a} \cdot P, \mathfrak{b} \cdot P) \mapsto \mathfrak{ab} \cdot P$ gives $\mathcal{X}$ an implicit group structure, and $\mu : (x_1, \ldots, x_r, y) \mapsto \left(\prod_i \mathfrak{g}_i^{x_i}\right) \cdot \mathfrak{a}^y \cdot P$ defines a homomorphism $\mathbb{Z}^r \times \mathbb{Z} \to \mathcal{X}$;
3. We can evaluate $(y, \mathfrak{a} \cdot P) \mapsto \mathfrak{a}^y \cdot P$, hence μ, using $\Theta(\log n)$ calls to PAR
4. Computing $\ker \mu = \{(x_1, \ldots, x_r, y) : \mathfrak{g}_1^{x_1} \cdots \mathfrak{g}_r^{x_r} \mathfrak{a}^y = 1_{\mathfrak{G}}\}$ is a hidden subgroup problem (Shor again);
5. Any $(a_1, \ldots, a_r, 1)$ in $\ker \mu$ gives a representation $\mathfrak{a} = \prod_i \mathfrak{g}_i^{a_i}$.

Classical separation between PAR and VEC

Curiously, in the **classical** setting we *don't* have PAR $\implies$ VEC.

For classical CDHP $\implies$ DLP we have a standard **black-box field** approach:

1. Reduce to prime order case (Pohlig–Hellman algorithm);
2. View $\mathfrak{G}$ as a representation of $\mathbb{F}_p$ via $\mathfrak{G} \ni \mathfrak{g}^a \leftrightarrow a \in \mathbb{F}_p$;
   - for $+$, use group operation $(\mathfrak{g}^a, \mathfrak{g}^b) \mapsto \mathfrak{g}^a \mathfrak{g}^b = \mathfrak{g}^{a+b}$
   - for $\times$, use $\mathfrak{G}$-DH oracle $(\mathfrak{g}, \mathfrak{g}^a, \mathfrak{g}^b) \mapsto \mathfrak{g}^{ab}$
3. den Boer, Maurer, Wolf, ...: conditional polynomial-time reduction.[1]

Does not work for PAR $\implies$ VEC, because PAR oracle $(P, \mathfrak{a} \cdot P, \mathfrak{b} \cdot P) \mapsto \mathfrak{ab} \cdot P$ only yields a group structure on $\mathcal{X}$, not a field structure.

[1] See the appendix for a quick sketch.

Hashing with isogeny graphs

The supersingular ℓ-isogeny graph

What about supersingular elliptic curves over $\mathbb{F}_{p^2}$?

The ℓ-isogeny graph of supersingular curves over $\mathbb{F}_{p^2}$ is

- a $(\ell+1)$**-regular connected** graph with $\approx p/12$ vertices;
- an **expander** graph (a Ramanujan graph!);
- random walks become uniformly distributed after $\approx \log p$ steps

The graph for $\ell = 2$ made its first appearance in algorithmic number theory with Mestre (1986), who used it to compute traces of the Hecke operator T_2.

The Charles–Goren–Lauter hash function

A **cryptographic hash function** maps long binary strings to compact values in such a way that finding preimages and collisions is computationally infeasible.

Charles, Goren, and Lauter (2009) proposed a **provably strong** hash function based on non-backtracking walks in the supersingular 2-isogeny graph:

1. Fix, once and forall, adjacent vertices j_{-1} and j_0 (the base point).
2. Choose a "sign" on $\mathbb{F}_{p^2}$.
3. For the i-th bit b_i in the input string $b_0 b_1 \cdots b_n$:
   - $\Phi_2(j_i, X) = (X - j_{i-1})(X - j_+)(X - j_-)$, where the roots j_+ and j_- are labelled using the sign in the quadratic formula (say);
   - we take $j_{i+1} := j_+$ if $b_i = 0$, or j_- if $b_i = 1$.
4. The output is j_n (mapped linearly into $\mathbb{F}_p$, to save space).

Pros and cons

The Charles–Goren–Lauter hash function is **extremely slow**.

In its original form, using modular polynomials to compute neighbouring vertices, you need to compute a square root in $\mathbb{F}_{p^2}$ to process each bit in the input!

(We can go faster using 2-torsion and explicit isogenies, but it is still *much* slower than everyday cryptographic hash functions...)

However, it *does* enjoy nice number-theoretic security proofs.

- We know the graph's spectral properties, diameter, etc.
- Finding collisions $\implies$ finding cycles, which are necessarily very large.

Takashima's hash function

Question: What happens in genus 2?

Takashima (2018) suggested generalizing the Charles–Goren–Lauter hash function using supersingular genus-2 Jacobians and Richelot isogenies:

- 2-isogenies are replaced with $(2, 2)$-isogenies
- The 3-regular 2-isogeny graph becomes a 15-regular $(2, 2)$-isogeny graph
- Ignore products of elliptic curves (extremely unlikely to hit one anyway)
- j-invariants are replaced with Igusa invariants

Takashima avoids backtracking, which means the hash function is driven by a base-14 encoding of the input string!

Not backtracking is not enough

Takashima's hash turns out to be easy to break: we can trivially construct cycles of length 4 anywhere (hence collisions).

(In the elliptic graph, we avoid backtracking to ensure that cycles have the expected length, and that any cycles are necessarily extremely long...)

In the genus-2 graph, the composition of two $(2, 2)$-isogenies $\psi \circ \phi$ can be

1. A $(2, 2, 2, 2)$-isogeny ($\psi \cong$ dual of ϕ; **backtracking**)
2. A $(4, 4)$-isogeny ($\ker \psi \cap \mathrm{im}\phi = 0$)
3. A $(4, 2, 2)$-isogeny ($\ker \psi \cap \mathrm{im}\phi \cong \mathbb{Z}/2\mathbb{Z}$)

...And the composition of two non-dual $(4, 2, 2)$-isogenies can be a $(4, 4, 4, 4)$-isogeny: that is, multiplication by 4 on the starting curve! This is a trivially generated cycle of length 4.

The superspecial graph

Castryck–Decru–S. (2019): repairing Takashima's hash function.

First: the correct graph is the **superspecial** graph, comprised of abelian surfaces that are *unpolarizedly isomorphic* to products of supersingular elliptic curves.

- the graph we start in anyway, and
  (in fact, we don't know how to construct a vertex outside this subgraph!)
- closed under $(2, 2)$-isogenies (or any separable isogenies).

We avoid not only backtracking, but any compositions giving $(4, 2, 2)$-isogenies: this leaves 8 steps forwards at each vertex.

Open questions

Our superspecial hash function is more funky[2] than the elliptic supersingular hash function.

The superspecial graph over $\mathbb{F}_{p^2}$ has size roughly p^3, which means that we can take p one-third the size of what we need for the elliptic graph. It's also a good opportunity to use efficient Kummer surface arithmetic.

But to prove any security properties, there are lots of things we need to know:

- Is the superspecial $(2,2)$-isogeny graph **connected**?
- What are its **expansion** and **mixing** properties like?
- What happens when we avoid $(4,2,2)$-isogenies?

[2] In the absence of a mathematical definition of funkiness, this statement is vacuously true.

Postquantum key exchange in the full supersingular graph

The problem with HHS-DH

Breaking isogeny HHS keypairs over $\mathbb{F}_q$ requires $O(q^{1/4})$ classical operations (Galbraith–Hess–Smart, Galbraith–Stolbunov).

In 2010, Childs, Jao, and Soukharev found an $L_N(1/2)$ quantum isogeny evaluation algorithm, which (combined with Kuperberg's abelian hidden shift algorithm) gives an $L_N(1/2)$ quantum attack on CRS.

This line of attack explicitly requires the action of a commutative group.

In 2011, Jao and De Feo proposed a key exchange based on composing isogenies, but with no hidden commutative group: **Supersingular Isogeny Diffie–Hellman**.

CRS keys A long series of short walks in cyclic graphs.

SIDH keys One long walk in an expander graph.

SIDH

Chose a prime $p = c2^m3^n - 1$. Supersingular $\mathcal{E}/\mathbb{F}_{p^2}$ have $\mathcal{E}(\mathbb{F}_{p^2}) \cong (\mathbb{Z}/c2^m3^n\mathbb{Z})^2$.
We choose a base curve $\mathcal{E}_0$ and bases $\langle P_2, Q_2\rangle = \mathcal{E}_0[2^m]$ and $\langle P_3, Q_3\rangle = \mathcal{E}_0[3^n]$.

Phase 1 Alice samples a secret $a \in \mathbb{Z}/2^m\mathbb{Z}$; computes the 2^m-isogeny $\phi_A : \mathcal{E}_0 \to \mathcal{E}_A := \mathcal{E}_0/\langle P_2 + aQ_2\rangle$; publishes $(\mathcal{E}_A, \phi_A(P_3), \phi_A(Q_3))$.

Bob samples a secret $b \in \mathbb{Z}/3^n\mathbb{Z}$; computes the 3^n-isogeny $\phi_B : \mathcal{E}_0 \to \mathcal{E}_B := \mathcal{E}_0/\langle P_3 + bQ_3\rangle$; publishes $(\mathcal{E}_B, \phi_B(P_2), \phi_B(Q_2))$.

Phase 2 Alice computes the 2^m-isogeny $\phi'_A : \mathcal{E}_B \to \mathcal{E}_{BA} := \mathcal{E}_B/\langle \phi_B(P_2) + a\phi_B(Q_2)\rangle$; derives the shared secret $S = j(\mathcal{E}_{BA})$.

Bob computes the 3^n-isogeny $\phi'_B : \mathcal{E}_A \to \mathcal{E}_{AB} := \mathcal{E}_A/\langle \phi_A(P_3) + b\phi_A(Q_3)\rangle$; derives the shared secret $S = j(\mathcal{E}_{AB})$.

SIDH keys

SIDH public key validation is extremely problematic.

Consider the path from the base curve P to Bob's public key B:

$$P \xrightarrow{3} B_1 \xrightarrow{3} \cdots\cdots\cdots\rightarrow \xrightarrow{3} B_{n-1} \xrightarrow{3} B_n = B$$

Suppose we have an oracle $V_3(X, Y, k)$ which returns `True` iff there is a 3^k-isogeny $X \to Y$; so B is valid if $V_3(P, B, n)$.

Compute the 4 neighbouring curves 3-isogenous to B_n (easy).
The curve C such that $V_3(P, C, n-1) = \texttt{True}$ is B_{n-1}.

Iterating, we unwind the path from P to B, revealing Bob's secret key.

See Galbraith–Petit–Shani–Ti (2016) for more details.

From SIDH to SIKE

SIDH key validation is dangerous, so we **cannot use SIDH** for static or non-interactive key exchange (**NIKE**).

The **Fujisaki–Okamoto transform** turns SIDH into an IND-CCA2 secure KEM, **SIKE**, which has been **submitted the NIST process**.

The optimized C/assembly implementation of SIKE aiming at NIST security level 1 runs in **about 10ms** on a PC.

Postquantum NIKE: HHS revisited

The search for a postquantum NIKE

By early 2017 there were plenty of postquantum **Key Encapsulation Mechanisms** (KEMs), but there was still **no drop-in replacement for classic DH**.

In particular: **no postquantum NIKE** (to replace static DH).

- SIDH comes closest to matching the DH API, but can't be used for static/non-interactive key exchange (no public key validation)
- SIKE is a safe KEM but doesn't match the API or do NIKE.
- Other postquantum paradigms (lattices, codes, multivariate, ...) offer high-speed KEMs, but no **exact** DH equivalent.

HHS-DH: the state of affairs

In theory, Couveignes–Rostovtsev–Stolbunov is a good candidate for postquantum DH/NIKE:

- it has the same API as classic Diffie–Hellan,
- key validation is just verifying an endomorphism ring (which is easy).

In practice, CRS seemed way too inefficient...

...But Luca De Feo, Jean Kieffer, and I decided to go back and try it again.

Towards practical commutative isogeny key exchange

De Feo–Kieffer–S. (Asiacrypt 2018): simple **algorithmic improvements** for HHS-DH.

- Faster ℓ-isogenies when kernel points are defined over $\mathbb{F}_{q^k}$ with $k \ll \sqrt{\ell}$
- Exploiting quadratic twists to eliminate quadratic extensions

Parameter selection: need (O_K, q) s.t. if $\mathcal{X} = \{\mathcal{E}/\mathbb{F}_q$ with CM by $O_K\}/\cong$, then

1. $\ell \mid \mathcal{E}(\mathbb{F}_{q^k})$ with k small as possible for many small split ℓ;
2. $\#\mathcal{X} = \#\mathrm{Cl}(O_K)$ is very large (ideally $\sim \sqrt{q}$), and
3. we can construct an $\mathcal{E} \in \mathcal{X}$ in polynomial time.

Fully optimizing 1 $\implies$ simultaneous control of O_K and q $\implies$ **kills 2 and/or 3**.

Best we can do: try random curves with an early-abort SEA, eliminating curves with no tiny-order points and/or bad endomorphism rings, hoping for good 1...

Result: PoC implementation at NIST security level 1 completes a **DH in 8 minutes**.

CSIDH: a NIKE less ordinary

CSIDH (pronounced **seaside**) = *Commutative* Supersingular Isogeny DH
Castryck–Lange–Martindale–Panny–Renes (Asiacrypt 2018).

A cute solution to our parameter problem: we need a quadratic imaginary endomorphism ring O_K, but *the isogeny class need not be ordinary!*

Use the **supersingular isogeny class** over $\mathbb{F}_p$, so $K = \mathbb{Q}(\sqrt{-p})$.

This gives us full control over requirements 1 and 2, with easy 3.
Choose p such that $p + 1 = c\prod_i \ell_i$ with all the small ℓ_i you want; we can easily construct supersingular curves over $\mathbb{F}_p$. Bonus: the twist trick always applies!

Result: PoC implementation at NIST security level 1 completes a **DH in 100ms**.

Class group structures

When CSIDH was proposed in 2018, it used the 511-bit prime

$$p = 4 \cdot \Big(\prod_{i=1}^{73} \ell_i\Big) \cdot 587 - 1 \quad \text{where } \ell_i = i\text{-th smallest odd prime}$$

The class group $\mathrm{Cl}(\mathbb{Z}[\sqrt{-p}])$ is crucial, but nobody knew its order!

This is an extremely rare case of a cryptographic protocol using a group whose structure is **unknown but not secret**. *No Pohlig–Hellman: it shouldn't matter?*

Buellens–Kleinjung–Vercauteren, May 2019 (record class group computation!):

$$\begin{aligned}\#\mathrm{Cl}(\mathbb{Z}[\sqrt{-p}]) = 3 \times 37 \times 1407181 \times 51593604295295867744293584889 \\ \times 31599414504681995853008278745587832204909\,.\end{aligned}$$

Genus-2 key exchange

Question: what happens to all this in genus 2?

Genus-2 SIDH: first steps by Ti and Flynn (2019).

- Plenty of practical improvements to be made
- Security depends on the same open questions that we have for the superspecial hash

Genus-2 CSIDH: looks hard!

- Computing higher-degree isogenies in genus 2 is feasible when you have a lot of 2-level structure (Cosset 2011, Lubicz and Robert, 2012), which we do...
- But the isogeny graph structure is, again, more complicated! Might make more sense to stick to abelian surfaces with RM by a class-number-1 ring.

Conclusion

Conclusions

- In CSIDH, isogeny-based crypto now has a **practical postquantum drop-in replacement** for Diffie–Hellman. *Other protocols are in progress.*
- Couveignes' **Hard Homogeneous Spaces** framework helps to model and analyse postquantum DH protocols on an abstract level, without understanding the mechanics of isogenies
- Pre- and post-quantum DH have the same "API",
  but **HHS-DH does not respect Group-DH intuitition**.
- **Genus 2** may give better algorithmic performance in some cases, but we need to know more about the **isogeny graphs** to guarantee security

References and appendices

The Maurer reduction: how does it work?

We want to **solve a DLP** instance $\mathfrak{h} = \mathfrak{g}^x$ in $\mathfrak{G}$ of prime order p, **given a DH oracle** for $\mathfrak{G}$ (so we can compute $\mathfrak{g}^{F(x)}$, $\forall$ poly F):

1. Find an $\mathcal{E}/\mathbb{F}_p$ s.t. $\mathcal{E}(\mathbb{F}_p)$ has **polynomially smooth order** and compute a generator (x_0, y_0) for $\mathcal{E}(\mathbb{F}_p)$.
   Pohlig–Hellman: solve DLPs in $\mathcal{E}(\mathbb{F}_p)$ in polynomial time.
2. Use Tonelli–Shanks to compute a $\mathfrak{g}^y$ s.t. $\mathfrak{g}^{y^2} = \mathfrak{g}^{x^3+ax+b}$.
   If this fails: replace $\mathfrak{h} = \mathfrak{g}^x$ with $\mathfrak{h}\mathfrak{g}^\delta = \mathfrak{g}^{x+\delta}$ and try again...
   Now $(\mathfrak{g}^x, \mathfrak{g}^y)$ is a point in $\mathcal{E}(\mathfrak{G})$; we still don't know x or y.
3. Solve the DLP instance $(\mathfrak{g}^x, \mathfrak{g}^y) = [e](\mathfrak{g}^{x_0}, \mathfrak{g}^{y_0})$ in $\mathcal{E}(\mathfrak{G})$ for e.
4. Compute $(x, y) = [e](x_0, y_0)$ in $\mathcal{E}(\mathbb{F}_p)$ and return x.

Finding the curve $\mathcal{E}/\mathbb{F}_p$ in Step 1 is the tricky part! It seems to work in practice for cryptographically useful p, even if it doesn't work in theory for arbitrary p.

Further reading: discrete logs in abelian varieties

- C. Diem: *An index calculus algorithm for plane curves of small degree* (ANTS 2006)
- G. Frey and E. Kani: *Correspondences on Hyperelliptic Curves and Applications to the Discrete Logarithm* (SIIS 2011)
- P. Gaudry: *Algorithmique des courbes algébriques pour la cryptologie* (HdR, Univ. Nancy I, 2008)
- P. Gaudry, E. Thomé, N. Thériault, and C. Diem: *A double large prime variation for small genus hyperelliptic index calculus* (Math. Comp. 2007)
- B. Smith: *Isogenies and the discrete logarithm problem in Jacobians of genus 3 curves* (Eurocrypt 2007)

Further reading: Kummer varieties in cryptography

- D. J. Bernstein: *Curve25519: New Diffie–Hellman speed records* (PKC 2006).
- D. J. Bernstein, C. Chuengsatiansup, T. Lange, and P. Schwabe: *Kummer strikes back: new DH speed records* (Asiacrypt 2014).
- C. Costello and B. Smith: *Montgomery curves and their arithmetic: the case of large characteristic fields* (J. Crypt. Eng 2017).
- J. Renes and B. Smith: *qDSA: Small and secure digital signatures with curve-based Diffie–Hellman pairs* (Asiacrypt 2017).

Further reading: basic algorithms for isogenies

- C. Delfs and S. D. Galbraith: *Computing isogenies between supersingular elliptic curves over* $\mathbb{F}_p$ (Des. Codes Cryptography 2016)
- S. D. Galbraith and A. Stolbunov: *Improved algorithm for the isogeny problem for ordinary elliptic curves* (App. Alg. Eng. Commun. Comput. 2002)
- D. R. Kohel: *Endomorphism rings of elliptic curves over finite fields* (PhD, Berkeley 1996)
- J.-F. Mestre: *La méthode des graphes. Exemples et applications* (Katata 1986)
- J. Vélu: *Isogénies entre courbes elliptiques* (C. R. Acad. Sci. Paris Sér. A-B, 1971)

Further reading: commutative isogeny-based crypto

- W. Buellens, T. Kleinjung, and F. Vercauteren: *CSI-FiSh: efficient isogeny based signatures through class group computations* (ePrint, 2019)
- W. Castryck, T. Lange, C. Martindale, L. Panny, and J. Renes: *CSIDH: an efficient post-quantum commutative group action* (Asiacrypt 2018).
- J.-M. Couveignes: *Hard homogeneous spaces* (ePrint, 2006).
- L. De Feo, J. Kieffer, and B. Smith: *Towards practical key exchange from ordinary isogeny graphs* (Asiacrypt 2018).
- S. D. Galbraith, L. Panny, B. Smith, and F. Vercauteren: *Quantum equivalence of the DLP And CDHP for group actions* (ePrint 2019)
- B. Smith: *Pre- and post-quantum Diffie–Hellman from groups, actions, and isogenies* (SAC 2018)
- A. Stolbunov: *Constructing public-key cryptographic schemes based on class group action on a set of isogenous elliptic curves* (Adv. Math. Commun. 2010)

Further reading: noncommutative isogeny-based crypto

- W. Castryck, T. Decru, and B. Smith: *Hash functions from superspecial genus-2 curves using Richelot isogenies* (NuTMiC 2019)
- D. X. Charles, E. Goren, and K. E. Lauter: *Cryptographic hash functions from expander graphs* (J. Crypt. 2009)
- A. M. Childs, D. Jao, and V. Soukharev: *Constructing elliptic curve isogenies in quantum subexponential time* (J. Math. Crypt 2010)
- S. D. Galbraith, C. Petit, C. Shani, and Y. B. Ti: *On the security of supersingular isogeny cryptosystems* (Asiacrypt 2016)
- D. Jao, L. De Feo, and J. Plût: *Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies* (J. Math. Crypt 2014)