Isogeny-based cryptography: cryptanalysis results

Christophe Petit

University of Birmingham
Isogeny-based cryptography

- Recently proposed for post-quantum cryptography
- Natural problems from a number theory point of view
- Some history, e.g. David Kohel’s PhD thesis in 1996
- Some still exponential time, even for quantum computers
Hard problems?

- Isogeny computation problem (CGL hash, CSIDH): Given two randomly chosen isogenous elliptic curves, compute an isogeny between them.

- Supersingular Isogeny Diffie-Hellman protocol: Let $p$ a prime such that $2^e 3^f | (p - 1)$ and $2^e \approx 3^f \approx \sqrt{p}$. Given two supersingular elliptic curves $E_0, E_1$ over $\mathbb{F}_p^2$ connected by an isogeny $\varphi: E_0 \to E_1$ of degree $2^e$, and given the action of $\varphi$ on the $3^f$-torsion, compute $\varphi$. 
Outline

Computing isogenies (generic supersingular case)

Supersingular isogeny key exchange protocol

Computing isogenies (ordinary curves and CSIDH)
Outline

Computing isogenies (generic supersingular case)

Supersingular isogeny key exchange protocol

Computing isogenies (ordinary curves and CSIDH)
Hash of the Future?

Have you ever struggled to solve a maze? Then imagine trying to find a path through a tangled, three-dimensional maze as large as the Milky Way. By incorporating such a maze into a hash function, Kristin Lauter of Microsoft Research in Redmond, Washington, is betting that neither you nor anyone else will solve that problem.

Technically, Lauter’s maze is called an “expander graph” (see figure, right). Nodes in the graph correspond to elliptic curves, or equations of the form $y^2 = x^3 + ax + b$. Each curve leads to three other curves by a mathematical relation, now called isogeny, that Pierre de Fermat discovered while trying to prove his famous Last Theorem.

To hash a digital file using an expander graph, you would convert the bits of data into directions: 0 would mean “turn right,” 1 would mean “turn left.” In the maze illustrated here, after the initial step 1-2, the blue path encodes the directions 1, 0, 1, 1, 0, 0, 0, 1, ending at point 24, which would be the digital signature of the string 101100001. The red loop shows a collision of two paths, which would be practically impossible to find in the immense maze envisioned by Lauter.

Although her hash function (developed with colleagues Denis Charles and Eyal Goren) is provably secure, Lauter admits that it is not yet fast enough to compete with iterative hash functions. However, for applications in which speed is less of an issue—for example, where the files to be hashed are relatively small—Lauter believes it might be a winner.

—D.M.
Charles-Goren-Lauter hash function (2)

- Suggested parameters:
  - Supersingular curves (for optimal expansion properties)
  - $\ell$ isogeny-graph with $\ell = 2$ (for efficiency)
  - “Special” starting curve $E_0$ (typically $j = 1728$) with known endomorphism ring
    (no convenient way to select a “random” starting curve)

- Collision, preimage, second preimage resistance naturally translate into isogeny problems, where isogeny degrees are required to be $\ell^e$ for some $e$
  - Preimage $\approx$ isogeny $\approx$ path between two vertices
  - Collision $\approx$ endomorphism $\approx$ cycle in the graph
The endomorphism ring of a supersingular curve

- The endomorphism ring of a supersingular curve is a maximal order in the quaternion algebra $B_{p,\infty}$
- In fact, Deuring correspondence [D31]: bijection from supersingular curves over $\mathbb{F}_p^2$ (up to Galois conjugacy) to maximal orders in $B_{p,\infty}$ (up to conjugation)

$$E \rightarrow O \approx \text{End}(E)$$

- Under this correspondence, an isogeny $\phi : E_0 \rightarrow E_1$ corresponds to a left ideal of $O_0 \approx \text{End}(E_0)$ which is also a right ideal of $O_1 \approx \text{End}(E_1)$
Strategy to break CGL hash function [PL17]

- Idea: given two curves $E_0$ and $E_1$
  1. Compute $\text{End}(E_0)$ and $\text{End}(E_1)$
  2. Translate collision and preimage resistance properties from the elliptic curve setting to the quaternion setting
  3. Break collision and preimage resistance for quaternions
  4. Translate the attacks back to elliptic curve setting

- Results so far (1)
  - Breaking CGL hash function (for randomly chosen $E_0$) is equivalent to computing endomorphism rings
Core problem : computing endomorphisms

- Kohel’s algorithm [K96] : fix a small $\ell$. Given a curve $E$, compute all its neighbors in isogeny graph. Compute all neighbors of neighbors, etc, until a loop is found, corresponding to an endomorphism.

- Complexity $\tilde{O}(\sqrt{p})$
Some variants

- To compute an isogeny between two curves, grow two trees until a collision is found
- Delfs-Galbraith [DG16]: first compute isogenies to two $\mathbb{F}_p$ curves, then connect those two curves
- Time-memory trade-offs (van Oorschot-Wiener) [vOW94]
- Quantum speedups (?): cube root quantum claw finding, but may not be practical [JS19]
Strategy to break CGL hash function [PL17]

- Idea: given two curves $E_0$ and $E_1$
  1. Compute $\text{End}(E_0)$ and $\text{End}(E_1)$
  2. Translate collision and preimage resistance properties from the elliptic curve setting to the quaternion setting
  3. Break collision and preimage resistance for quaternions
  4. Translate the attacks back to elliptic curve setting

- Results so far (1)
  - Breaking CGL hash function (for randomly chosen $E_0$) is equivalent to computing endomorphism rings
Strategy to break CGL hash function [PL17]

- Idea: given two curves $E_0$ and $E_1$
  1. Compute $\text{End}(E_0)$ and $\text{End}(E_1)$
  2. Translate collision and preimage resistance properties from the elliptic curve setting to the quaternion setting
  3. Break collision and preimage resistance for quaternions
  4. Translate the attacks back to elliptic curve setting

- Results so far (2)
  - 2-4 can be solved in polynomial time (modulo heuristics)
  - When a “special” $E_0$ is chosen in CGL hash function, we can compute collisions in polynomial time
  - Explicit Deuring correspondence easy in one direction (given $O$ compute corresponding $j$)
Key tools

- Algorithms to solve quaternion norm equations [KLPT14]
- Translation between quaternion ideals and isogenies [W69]
  - Let $E_0$ with known $\text{End}(E_0) \approx O_0 \subset B_{p,\infty}$
  - Isogenies from $E_0$ correspond to left ideals of $O_0$
  - Correspondence computed by identifying kernels
  - Efficient for powersmooth norms/degrees
- “Quaternion $\ell$-isogeny algorithm” [KLPT14,GPS17]
  - Replace ideal by equivalent one with powersmooth norm
Partial attack on CGL hash function [PL17]

- Suppose CGL hash function uses a special curve $E_0$
- Goal: compute an endomorphism of $E_0$ of degree $\ell^e$
  (this gives a collision with the void message)
- Compute $\alpha \in O_0 \approx \text{End}(E_0)$ of norm $\ell^e$ (as in [KLPT14])
- Deduce a collision path in the quaternion setting
  $I_i = O_0\ell^i + O_0\alpha$, $i = 1, \ldots, e$, where $n(I_i) = \ell^i$
- For each $i$
  - Compute $J_i \approx I_i$ with powersmooth norm
  - Compute corresponding isogeny $\varphi_i : E_0 \rightarrow E_i$
- Deduce a collision path $(E_0, E_1, \ldots, E_e = E_0)$
Strategy to break CGL hash function [PL17]

- Idea: given two curves $E_0$ and $E_1$
  1. Compute $\text{End}(E_0)$ and $\text{End}(E_1)$
  2. Translate collision and preimage resistance properties from the elliptic curve setting to the quaternion setting
  3. Break collision and preimage resistance for quaternions
  4. Translate the attacks back to elliptic curve setting

- Results so far
  - Breaking CGL hash function for randomly chosen $E_0$ is equivalent to computing endomorphism rings
  - When a “special” $E_0$ is chosen in CGL hash function, we can compute collisions in polynomial time
Outline

Computing isogenies (generic supersingular case)

Supersingular isogeny key exchange protocol

Computing isogenies (ordinary curves and CSIDH)
Diffie-Hellman key agreement

- Choose $g$ generating a cyclic group
- Alice picks a random $a$ and sends $g^a$
- Bob picks a random $b$ and sends $g^b$
- Alice computes $(g^b)^a = g^{ab}$
- Bob computes $(g^a)^b = g^{ab}$
- Eve cannot compute $a$, $b$ or $g^{ab}$ from $g^a$ and $g^b$
  (discrete logarithm, Diffie-Hellman problems)
Choose a prime \( p \), and \( N_A, N_B \in \mathbb{N} \) with \( \gcd(N_A, N_B) = 1 \)
Choose \( E_0 \) a supersingular curve over \( \mathbb{F}_{p^2} \)
Alice picks a cyclic subgroup \( G_A \subset E_0[N_A] \) defining an isogeny \( \phi_A : E_0 \to E_A = E_0/G_A \) and she sends \( E_A \) to Bob
Bob picks a cyclic subgroup \( G_B \subset E_0[N_B] \) defining an isogeny \( \phi_B : E_0 \to E_B = E_0/G_B \) and he sends \( E_B \) to Alice

\[
\begin{align*}
E_A &= E_0/G_A \\
E_B &= E_0/G_B
\end{align*}
\]

Shared key is \( E_0/\langle G_A, G_B \rangle = E_B/\phi_B(G_A) = E_A/\phi_A(G_B) \)
Isogeny-based Diffie-Hellman (2)

To compute the shared key Alice will need $\phi_B(G_A)$. This is achieved as follows:

- Let $G_A = \langle \alpha_A P_A + \beta_A Q_A \rangle$ where $\langle P_A, Q_A \rangle = E_0[N_A]$ and at least one of $\alpha_A$, $\beta_A$ coprime to $N_A$.
- Bob reveals $\phi_B(P_A)$ and $\phi_B(Q_A)$ in addition to $E_B$.
- Alice computes $\phi_B(G_A) = \langle \alpha_A \phi_B(P_A) + \beta_A \phi_B(Q_A) \rangle$.

- Can represent $\phi_A$ efficiently if $N_A$ smooth.
- Can represent torsion points efficiently if either
  - $N_A \mid p - 1$
  - $N_A = \prod \ell_i^{e_i}$ with $\ell_i^{e_i}$ small.
Supersingular key agreement protocol [JdF11]

\[
\begin{align*}
E_0/\langle R_A \rangle & \xrightarrow{\phi_A} E_0/\langle R_B \rangle \\
\phi_A(P_B), \phi_A(Q_B) & \xrightarrow{\phi_B} \phi_B(P_A), \phi_B(Q_A) \\
\phi_A(R_B) & \xrightarrow{\phi_B'} \phi_B(R_A) \\
E_0/\langle R_B \rangle & \xrightarrow{\phi_B} E_0/\langle R_A, R_B \rangle \\
\phi_B(P_A), \phi_B(Q_A) & \xrightarrow{\phi_A'} \phi_A(R_A)
\end{align*}
\]

- Jao-De Feo chose \( N_i = \ell_i^{e_i} \) and \( p = N_A N_B f + 1 \)
- A priori safer to use arbitrary primes and \( N_i \approx p^2 \)
Special isogeny problems

- In Jao-De Feo-Plût protocols special problems are used
  1. A special prime $p$ is chosen so that $p = N_1N_2 \pm 1$ with $N_1 \approx N_2 \approx \sqrt{p}$
  2. There are $\approx p/12$ supersingular invariants but only $N_1 \approx \sqrt{p}$ possible choices for $E_1$
  3. **Extra information provided**: compute $\phi : E_0 \rightarrow E_1$ of degree $N_1$ knowing $\phi(P)$ for all $P \in E_0[N_2]$

- Point 2 improves tree-based attacks to $O(p^{1/4})$
  (and similar improvements using van Oorschot-Wiener)
- We now focus on Point 3
Impact of torsion points

- Attack on Jao-De Feo-Plût protocol: compute an isogeny $\phi_1 : E_0 \rightarrow E_1$ of degree $N_1$ given action of $\phi_1$ on $E_0[N_2]$

- How useful is this additional information?
  - If $d = \gcd(N_1, N_2) \neq 1$ we can recover (part of) $\phi_1$
    - Write $\phi_1 = \phi_1' \circ \phi_d$ with $\deg \phi_d = d$
    - Solve DLP modulo $d$ to recover $\ker \phi_d$ hence $\phi_d$
    - Find $\phi_1'$ with a meet-in-the-middle approach
  - In SIDH we have $\gcd(N_1, N_2) = 1$ by design

- Useless?
Outline

Computing isogenies (generic supersingular case)

Supersingular isogeny key exchange protocol
  Using torsion points : active attacks
  Using torsion points : passive attacks

Computing isogenies (ordinary curves and CSIDH)
Active attacks on static keys (1)

- Attack idea: trick Alice into computing $\phi_A(P_B), \phi_A(Q_B)$ for $P_B, Q_B$ of order $N_A$ instead of $N_B$
  - $P_B, Q_B$ part of a (maliciously generated) public key
  - Fault attack during Alice’s computation [T17,GW17]

- More generally, $P_B, Q_B$ of order not coprime with $N_B$

- Countermeasure: check that $P_B, Q_B$ have order $N_B$
Active attacks on static keys (2) [GPST16]

- Attack model
  - Alice is using a static key $\alpha$ defining a cyclic subgroup $G_A = \langle P_A + \alpha Q_A \rangle \subseteq E_0[N_A]$
  - Instead of sending $\phi_B(P_A), \phi_B(Q_A)$ as expected, Bob adaptively chooses and sends $P_i, Q_i$
  - Bob learns whether this modifies the shared key $j(E_B/\langle P_i + \alpha Q_i \rangle) = j(E_B/\langle \phi_B(P_A) + \alpha \phi_B(Q_A) \rangle)$
  - Bob progressively recovers $\alpha$ with several $P_i, Q_i$

- Additional constraint: make sure $P_i, Q_i$ look as expected
  - $N_B P_i = N_B Q_i = O$
  - $e_{N_B}(P_i, Q_i) = e_{N_B}(\phi_A(P_B), \phi_A(Q_B)) = e_{N_B}(P_B, Q_B)^{N_A}$
Solution and countermeasure

- Solution for $N_A = 2^e$ :
  - Let $\alpha = A_i + 2^i \alpha'$
  - Replace $\phi_A(P_B), \phi_A(Q_B)$ by $P_i, Q_i$ such that

  \[
  \begin{pmatrix}
  P_i \\
  Q_i \\
  \end{pmatrix} = \frac{1}{\lambda_i} \begin{pmatrix}
  1 & -2^{n-i-1} A_i \\
  0 & 1+2^n - i - 1 \\
  \end{pmatrix} \begin{pmatrix}
  \phi_A(P_B) \\
  \phi_A(Q_B) \\
  \end{pmatrix}
  \]

  where $\lambda_i^2 = 1 + 2^n - i - 1 \mod 2^n$

  - We then have $\langle P_i + \alpha Q_i \rangle = \langle \phi_A(P_B) + \alpha \phi_A(Q_B) \rangle$ iff

  $2^{n-i-1} (-A_i + \alpha) \mod 2^n$

- Countermeasure : Fujisaki-Okamoto transform
  (factor 2 slowdown)
Outline

Computing isogenies (generic supersingular case)

Supersingular isogeny key exchange protocol
  Using torsion points : active attacks
  Using torsion points : passive attacks

Computing isogenies (ordinary curves and CSIDH)
Impact of torsion points

- Attack on Jao-De Feo-Plût protocol: compute an isogeny \( \phi_1 : E_0 \to E_1 \) of degree \( N_1 \) given action of \( \phi_1 \) on \( E_0[N_2] \).
- How useful is this additional information?
  - If \( d = \gcd(N_1, N_2) \neq 1 \) we can recover (part of) \( \phi_1 \), but in SIDH we have \( \gcd(N_1, N_2) \neq 1 \) by design.
  - Some active attacks can exploit torsion points.
- What about passive attacks? (honest users)
Warm-up : computing endomorphisms with auxiliary information

- Let $p$ be a prime and let $E$ be a supersingular elliptic curve defined over $\mathbb{F}_{p^2}$. Let $\phi$ be a non scalar endomorphism of $E$ with smooth order $N_1$. Let $N_2$ be a smooth integer with $\gcd(N_1, N_2) = 1$, and let $P, Q$ be a basis of $E[N_2]$.

- Let $R$ be a subring of $\text{End}(E)$ that is either easy to compute, or given (for example, scalar multiplications).

- Given $E, P, Q, \phi(P), \phi(Q), \deg\phi, R$, compute $\phi$.

- Best previous algorithm : meet-in-the-middle in $\tilde{O}(\sqrt{N_1})$
Algorithm sketch (with $R = \mathbb{Z}$)

- We know $\phi$ on the $N_2$ torsion.
  Deduce $\hat{\phi}$ on the $N_2$ torsion and $\text{Tr}(\phi)$ if $N_2 > 2\sqrt{N_1}$.
- Consider $\psi := a\phi + b$ for $a, b \in \mathbb{Z}$.
  Can evaluate $\psi$ on the $N_2$ torsion.
- Find $a, b \in \mathbb{Z}$ such that
  \[ \deg \psi = a^2 \deg \phi + b^2 + ab \text{Tr}\phi = N_2 N'_1 \]
  with $N'_1$ small and smooth. Write $\psi = \psi_{N'_1} \psi_{N_2}$.
- Identify $\ker \psi_{N_2}$ from $\psi(E[N_2])$ and deduce $\psi_{N_2}$.
- Find $\psi_{N'_1}$ with a meet-in-the-middle strategy.
- Find $\ker \phi$ by evaluating $(\psi - b)/a$ on the $N_1$ torsion, and deduce $\phi$. 
Finding \((a, b)\) and Complexity

- We have \(\deg \psi = a^2 \deg \phi + b^2 + ab \text{Tr} \phi\)
  
  
  \[
  = \left( b + a \frac{\text{Tr} \phi}{2} \right)^2 + a^2 \left( \deg \phi - \left( \frac{\text{Tr} \phi}{2} \right)^2 \right)
  \]

- We want \(\deg \psi = N_2 N_1'\) and \(N_1'\) small and smooth

- Solutions to \(\deg \psi = 0 \mod N_2\) form a dimension 2 lattice

- We compute a reduced basis, then search for a small linear combination of short vectors until \(N_1'\) smooth

- Heuristic analysis shows we can expect \(N_1' \approx \sqrt{N_1}\).

  Revealing \(\phi(E[N_2])\) leads to a near square root speedup.

  (Some parameter restrictions apply.)
Computing isogenies with auxiliary information

- Let $p$ be a prime. Let $N_1, N_2 \in \mathbb{Z}$ coprime. Let $E_0$ be a supersingular elliptic curve over $\mathbb{F}_{p^2}$. Let $\phi_1 : E_0 \to E_1$ be an isogeny of degree $N_1$.
- Let $R_0, R_1$ be subrings of $\text{End}(E_0), \text{End}(E_1)$ respectively. Assume $R_0$ contains more than scalar multiplications.
- Given $N_1, E_1, R_0, R_1$ and the image of $\phi_1$ on the whole $N_2$ torsion, compute $\phi_1$.
- Best previous algorithm: meet-in-the-middle in $\tilde{O}(\sqrt{N_1})$
General idea

- For $\theta \in \text{End}(E_0)$ consider $\phi = \phi_1 \theta \hat{\phi}_1 \in \text{End}(E_1)$
- Evaluate $\phi$ on the $N_2$ torsion
- Apply techniques from above on $\phi$
- Compute $\ker \phi \cap E_1[N_1]$
- Deduce $\ker \hat{\phi}_1$, then $\hat{\phi}_1$ and $\phi_1$
Remarks

➤ Several authors have suggested to use $j(E_0) = 1728$ for efficiency reasons. In this case $\text{End}(E_0)$ is entirely known and moreover it contains a degree 1 non scalar element $\theta$. Both aspects are useful in attacks.

➤ The paper develops two attacks but we expect variants and improvements to come.
Impact on Key Agreement Protocol

- For $j(E_0) = 1728$ and when $N_1 \approx p$ and $N_2 \approx N_1^4$, this approach leads to polynomial time key recovery (heuristic analysis)
- Assuming only that $\text{End}(E_0)$ has a small element, then if $\log N_2 \approx (\log^2 N_1)$, a variant of the above strategy also leads to polynomial time key recovery (heuristic analysis)
- Parameters suggested by De Feo-Jao-Plût $N_1 \approx N_2 \approx \sqrt{p}$ are not affected so far
Outline

Computing isogenies (generic supersingular case)

Supersingular isogeny key exchange protocol

Computing isogenies (ordinary curves and CSIDH)
Endomorphism ring computation

- Ordinary case: subexponential time (Bisson-Sutherland)

- CSIDH [CLMPR18]: we have

\[ \mathbb{Z}[\pi] \subseteq \text{End}(E) \subseteq \mathbb{Z}\left[\frac{\pi + 1}{2}\right] \]

where \( \pi : (x, y) \rightarrow (x^p, y^p) \)

and we can easily verify whether \( \frac{\pi + 1}{2} \in \text{End}(E) \)
Computing isogenies : classical algorithms

- We expect $O(p^{1/2})$ supersingular curves over $\mathbb{F}_p$
- Meet-in-the-middle approach restricted to these curves has cost $O(p^{1/4})$
Computing isogenies: quantum algorithms

- Reduction to hidden shift problem: let $E_0, E_1$ two isogenous curves. For $s \in \mathcal{C}_\ell(\text{End}(E_0))$ such that $E_1 = s \ast E_0$ we have

$$f(x) = g(xs)$$

where $f(x) = x \ast E_1$ and $g(x) = x \ast E_0$

- Kuperberg’s quantum algorithm (or variants) solves this in subexponential time
SIDH vs CSIDH?

- **CSIDH**
  - No torsion points revealed
  - Subexponential quantum attacks
  - Still exponential classical attacks

- **SIDH**
  - Torsion points revealed
    (leading to attacks on overstretched parameters)
  - Still exponential classical and quantum attacks
Outline

Computing isogenies (generic supersingular case)

Supersingular isogeny key exchange protocol

Computing isogenies (ordinary curves and CSIDH)
Conclusion

- Endomorphism ring computation & pure isogeny problems are natural problems with some history
- Still, we need more classical and quantum cryptanalysis, especially on problem variants
- SIDH or CSIDH? depends on two hypothetical threats
  - Improved torsion point attacks
    (or more attacks using further specificities in SIDH)
  - Devastating subexponential quantum attacks
Thanks!

- Questions?
Bibliography

- [D31] Deuring. Die Typen der Multiplikatorenringe elliptischer Funktionenkörper.
Bibliography

- [vOW94] van Oorschot-Wiener. *Parallel collision search with application to hash functions and discrete logarithms.*
- [JS19] Jaques-Schanck. *Quantum cryptanalysis in the RAM model : Claw finding attacks on SIKE.*
Bibliography

- [BS11] Bisson-Sutherland. *Computing the endomorphism ring of an ordinary elliptic curve over a finite field.*
- [CLMPR18] Castryck-Lange-Martindale-Panny-Rennes *CSIDH : An Efficient Post-Quantum Commutative Group Action*