Supersingular Isogeny Graphs in Cryptography

Kristin Lauter – Facebook AI Research

Park City Mathematics Institute

2021 Online Series—Lecture 1

July 26, 2021

Cryptography:

- The science of keeping secrets!
- But more than that…
  - Confidentiality
  - Authenticity
- Tools:
  - Encryption/Decryption
  - Digital signatures
  - Key exchange

Public Key Cryptography

- Key exchange: two parties agree on a common secret using only publicly exchanged information
- Signature schemes: allows parties to authenticate themselves
- Encryption: preserve confidentiality of data

- Examples of public key cryptosystems:

  RSA, Diffie-Hellman, ECDH, DSA, ECDSA

Applications:

- Secure browser sessions (https: SSL/TLS)
- Signed, encrypted email (S/MIME)
- Virtual private networking (IPSec)
- Authentication (X.509 certificates)

Elliptic Curve Cryptography

- *p* a large prime of cryptographic size
- Elliptic Curve defined by short Weierstrass equation:

$$E_1 : y^2 = x^3 + ax + b$$

- Labeled by j-invariants: *isomorphism invariant over $F_p$bar*

$$j(E_1)= 1728*4a^3/(4a^3+27b^2)$$

- Algebraic group with group law (chord and tangent method)
- *Supersingular* elliptic curves modulo p: *no p-torsion points over $F_p$bar*
  Isomorphism class has a representative defined over GF(p^2) (or GF(p))
  Endomorphism ring isomorphic to maximal order in definite quaternion algebra

Public Key Cryptography deployed today:

Security is based on hard math problems:

- Factoring large integers
- Discrete logarithm problem in *(Z/pZ)**
- Discrete logarithm problem in elliptic curve groups
- Weil pairing on elliptic curves

What do we mean by "hard" math problem?

Input represented by *m* bits:

Then the best known attack on the system runs in

exponential time in *m*	$O(2^m)$
sub-exponential time in *m*	$O(e^{c*m^\wedge 1/3\ (\log m)^\wedge 2/3})$
polynomial time in *m*	*O(polynomial in m)*

Example: to factor $n = p*q$ where $m = \log n$,

trial division takes *exponential time*

The Quantum threat:

Polynomial time Quantum algorithms for attacking current systems!

m = # bits

- Shor's algorithm for factoring $4m^3$ time and 2m qbits
- ECC attack requires $360m^3$ time and 6m qbits

  (Proos-Zalka, 2004)

Conclusion:

- RSA: m = 2048
- Discrete log m = 2048
- Elliptic Curve Cryptography m = 256 or 384

are not resistant to quantum attacks once a quantum computer exists at scale!

Timeline for Elliptic Curve Cryptography

- (2006) Suite B set requirements for the use of Elliptic Curve Cryptography
- (2016) CNSA requirements increase the minimum bit-length for ECC from 256 to 384. Advises that adoption of ECC not required.
- (2017) NIST international competition to select post-quantum solutions: PQC Competition

Post-quantum cryptography

Submissions to the NIST PQC competition based on hard math problems:

- Code-based cryptography (McEliece 1978)
- Multivariate cryptographic systems (Matsumoto-Imai, 1988)
- Lattice-based cryptography (Hoffstein-Pipher-Silverman, NTRU 1996)
- Supersingular Isogeny Graphs (Charles-Goren-Lauter 2005)

- Challenge! Need to see if these new systems are resistant to *both* classical and quantum algorithms!

Supersingular Isogeny Graphs

New hard problem introduced in 2005: [Charles-Goren-Lauter]

- *Finding paths between nodes in a Supersingular Isogeny Graph*

Graphs: G = (V, E) = (vertices, edges)

- k-regular, undirected graphs, with optimal expansion
- No known efficient routing algorithm

Application: Cryptographic Hash functions

A *hash function* maps bit strings of some finite length to bit strings of some fixed finite length

$$h : \{0,1\}^n \rightarrow \{0,1\}^m$$

- easy to compute
- unkeyed (do not require a secret key to compute output)
- Collision resistant
- Uniformly distributed output

Collision-resistance

- A hash function *h* is *collision resistant* if it is computationally infeasible to find two distinct inputs, *x, y*, which hash to the same output

$$h(x) = h(y)$$

- A hash function h is *preimage resistant* if, given any output of h, it is computationally infeasible to find an input, x, which hashes to that output.

Application: cryptographic hash function [CGL'06]

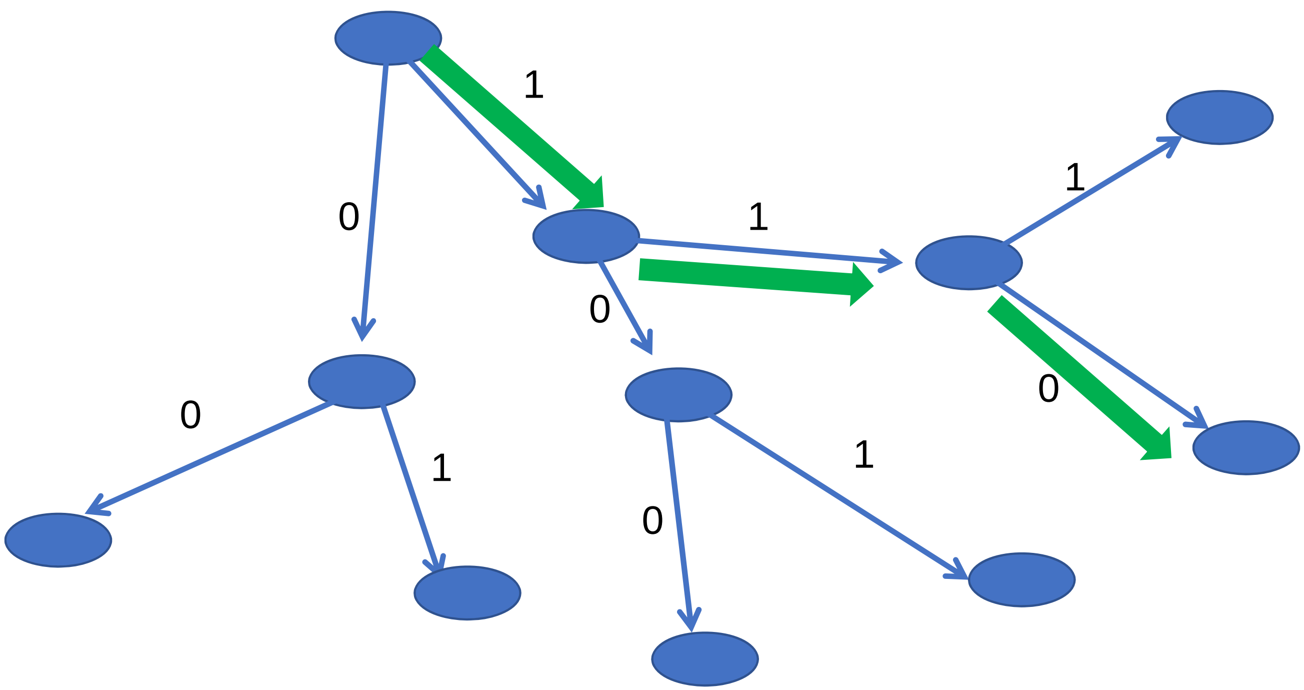
- k-regular graph G
- Each vertex in the graph has a label

Input: a bit string

- Bit string is divided into blocks
- Each block used to determine which edge to follow for the next step in the graph
- No backtracking allowed!

Output: label of the final vertex of the walk

Walk on a graph: 110



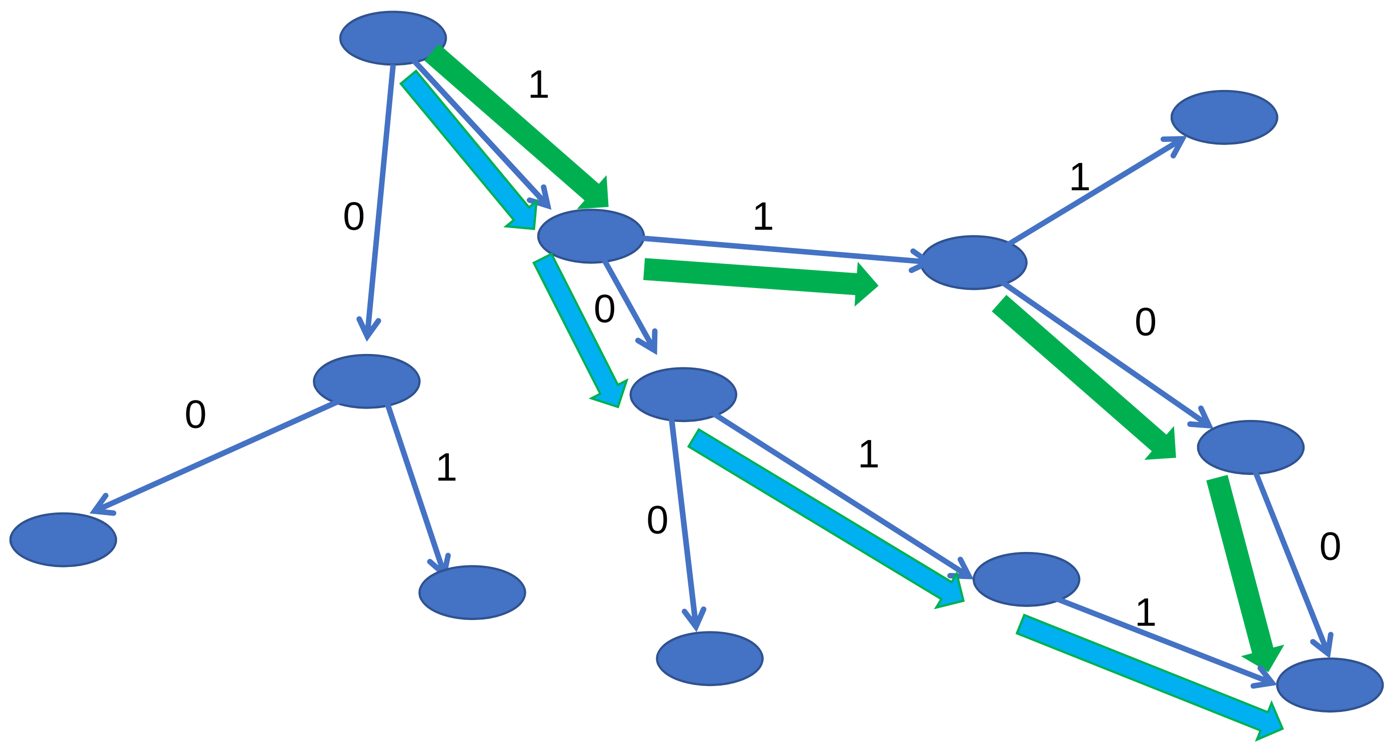
Simple idea

- Random walks on *expander* graphs are a good source of pseudo-randomness
- Are there graphs such that finding collisions is hard? (i.e. finding distinct paths between vertices is hard)
- Bad idea: hypercube (routing is easy, can be read off from the labels)

What kind of graph to use?

- Random walks on *expander* graphs mix rapidly: ~log(p) steps to a random vertex, p ~ #vertices
- *Ramanujan* graphs are optimal expanders
- To find a collision: *find two distinct walks of the same length which end at same vertex*

Colliding walks: 1100 and 1011



Graph of supersingular elliptic curves modulo p with isogeny edges (Pizer/Mestre graphs)

- Vertices: supersingular elliptic curves mod p
  - Curves are defined over GF(p^2) (or GF(p))

- Labeled by j-invariants
  - $E_1 : y^2 = x^3 + ax + b$
  - $j(E_1)= 1728*4a^3/(4a^3+27b^2)$

- Edges: Isogenies between elliptic curves

Supersingular Isogeny Graphs: vertices

Beautiful theorems in number theory:

- Deuring's correspondence

  Vertices $\leftrightarrow$ maximal orders in a quaternion algebra

  E $\leftrightarrow$ End(E)
- Eichler class number:

  # vertices ~ p/12

Supersingular Isogeny Graphs: edges

- Edges: degree ℓ isogenies between elliptic curves
  - k = ℓ+1 – regular
  - Undirected if we assume p == 1 mod 12
  - Graph is Ramanujan (Deligne, …)

Isogenies

- The degree of a separable isogeny is the size of its kernel
- To construct an ℓ -isogeny from an elliptic curve E to another, take a subgroup-scheme C of size ℓ, and take the quotient E/C.
- Formula for the isogeny and equation for E/C were given by Velu.

One step of the walk: (ℓ=2)

$E_1 : y^2 = x^3 + ax + b$

- $j(E_1)=1728*4a^3/(a^3+27b^2)$
- 2-torsion point $Q = (r, 0)$

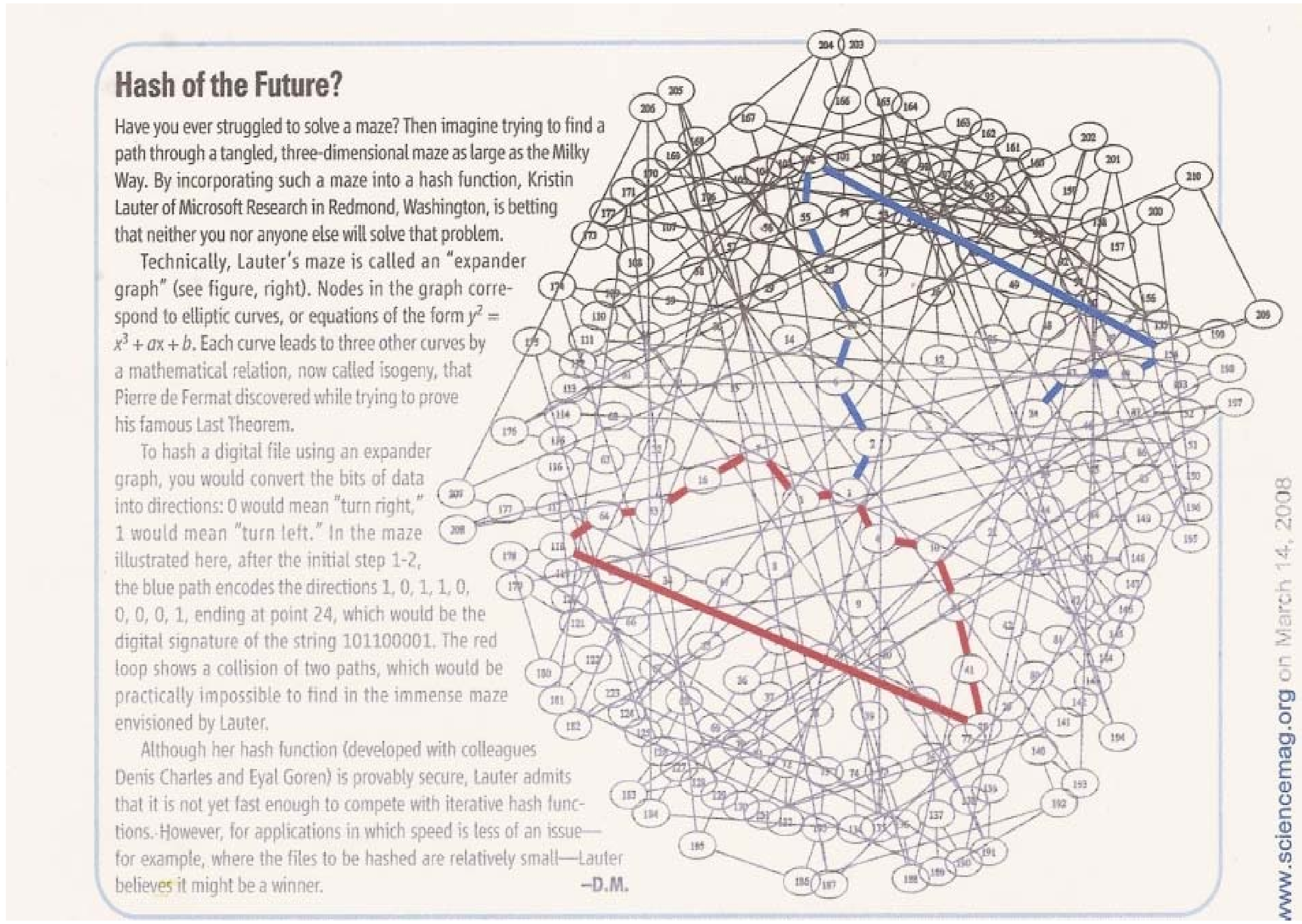
$E_2 = E_1 /Q$ (quotient of groups)

- $E_2 : y^2 = x^3 - (4a + 15r^2)x + (8b - 14r^3)$.

$E_1 \rightarrow E_2$

$(x, y) \rightarrow (x +(3r^2 + a)/(x-r), y - (3r^2 + a)y/(x-r)^2)$

Science magazine 2008



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, 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.

Hash of the Future?

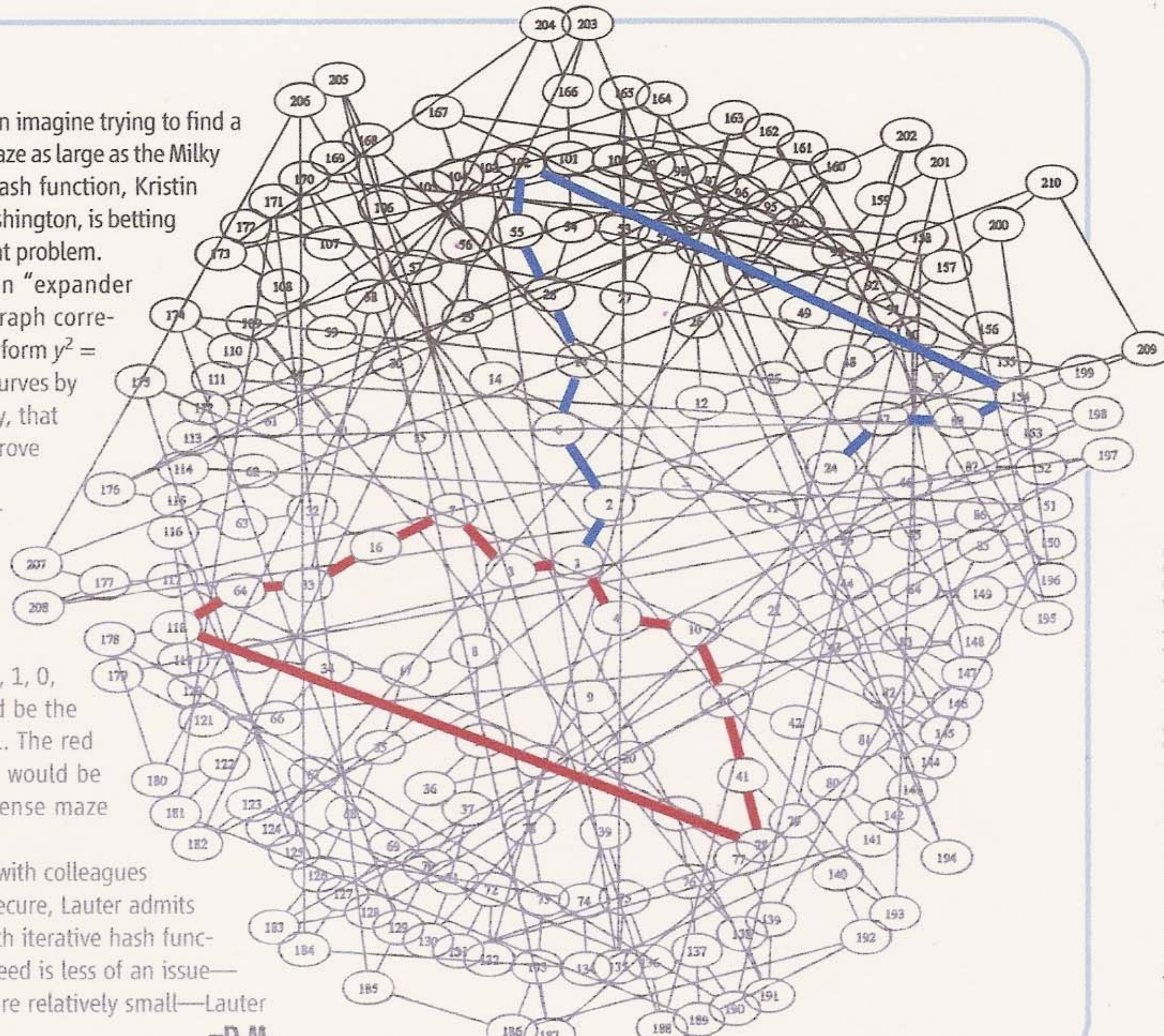
Have you ever struggled to solve a maze? Then imagine trying to find a path through a tangled, three-dimensional maze as large as the Milky Way. By incorporating such a maze into a hash function, Kristin Lauter of Microsoft Research in Redmond, Washington, is betting that neither you nor anyone else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



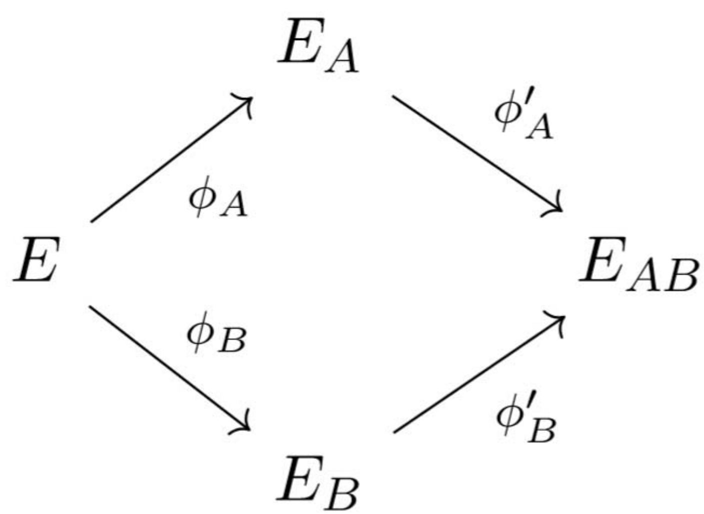
History

- Charles-Goren-Lauter presented at NIST 2005 competition,
  - IACR eprint 2006, published J Crypto 2009
- Later in 2006, two papers on eprint, never published:
  - Couveignes, ordinary case (Hard Homogeneous Spaces)
  - Rostovtsev-Stolbunov, ordinary case (Encryption)
- Ordinary case is very different for many reasons:
  - Volcano structure of graph
  - Action of an abelian class group

Applications of SIG

- Proposed as basis for other cryptosystems:
  - Key exchange: Jao-De Feo 2011
    (adds transmitting torsion images)
  - Encryption: De Feo-Jao-Plut, 2014
  - Signatures: Galbraith-Petit-Silva 2016

Key Exchange [Jao-DeFeo-Plut'11]



Key Exchange set-up

E: supersingular elliptic curve over GF(p^2)

p = $\ell_A^m \ \ell_B^n + 1$

ℓ_A and ℓ_B distinct primes (e.g. $\ell_A = 2$ and $\ell_B = 3$)

A and B want to exchange a key.

Public parameters:

A picks P_A, Q_A such that $< P_A, Q_A > = E[\ell_A^m]$

B picks P_B, Q_B such that $< P_B, Q_B > = E[\ell_B^n]$

Key Exchange (continued)

Secret parameters:

A picks two random integers m_A, n_A

A uses Velu's formulas to compute the isogeny

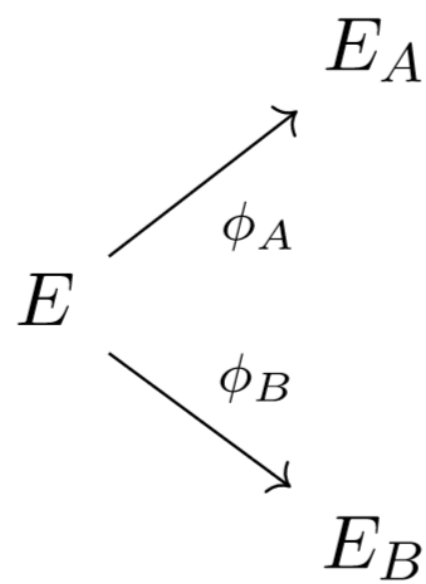
$$\varphi_A : E \longrightarrow E_A := E / < m_A P_A + n_A Q_A >$$

B picks two random integers m_B, n_B

B uses Velu's formulas to compute the isogeny

$$\varphi_B : E \longrightarrow E_B := E / < m_B P_B + n_B Q_B >$$

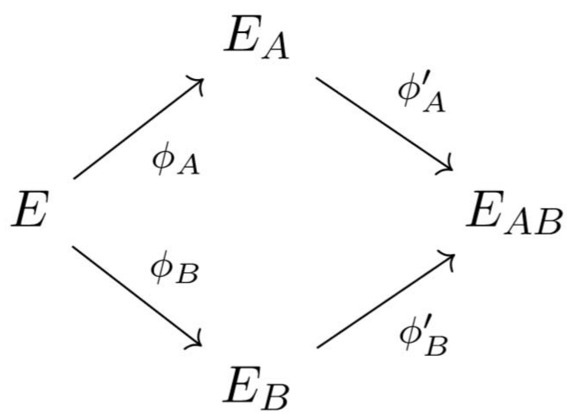
A and B have constructed the following diagram.



To complete the diamond, A and B exchange information:

A computes the points $\varphi_A(P_B)$ and $\varphi_A(Q_B)$ and sends $\{\varphi_A(P_B), \varphi_A(Q_B), E_A\}$ to B

B computes the points $\varphi_B(P_A)$ and $\varphi_B(Q_A)$ and sends $\{\varphi_B(P_A), \varphi_B(Q_A), E_B\}$ to A



The j-invariant of the curve E_{AB} is the shared secret.

Security of Key Exchange: relies on CGL path-finding problem

- If you can find the path between E and E_{A},
  then you can break the Key Exchange.
- Note that the walks on each stage of the Key Exchange protocol are of length roughly ½ the diameter!
- Thus the probability that there exists a path between any 2 nodes is roughly $p^{(-1/2)}$
- So if you can find any path, it is overwhelming likely to be the path used in the Key Exchange.

Reduction result from WIN4 project 2017

[Costache-Feigon-Lauter-Massierer-Puskas]

Theorem 5.3 *[CFLMP18] Assume as for the Key Exchange set-up that* $p = \ell_A^n \cdot \ell_B^m + 1$ *is a prime of cryptographic size, i.e.* $\log(p) \geq 256$, ℓ_A *and* ℓ_B *are small primes, such as* $\ell_A = 2$ *and* $\ell_B = 3$, *and* $n \approx m$ *are approximately equal. Given an algorithm to solve Problem 3.1 (Path-finding), it can be used to solve Problem 3.2 (Key Exchange) with overwhelming probability. The failure probability is roughly*

$$\frac{\ell_A^n + \ell_A^{n-1}}{p} \approx \frac{\sqrt{p}}{p}.$$

Other graphs

- Vary the isogeny degree
- Lubotzky-Phillips-Sarnak graph
  - Cycles found: Eurocrypt 2008, Zemor-Tillich
  - Preimages found: SCN 2008, Petit-Quisquater-Lauter
- Morgenstern graph, [Petit-Quisquater-Lauter 08]
- Higher dimensional analogues
  - Superspecial abelian surfaces [Charles-Goren-L 07]

“Isogenies in Cryptography” ongoing work:

- Alternate graphs/protocols:
  - CSIDH: Castryck-Lange-Martindale-Panny-Renes
- Dimension 2 analogues:
  - Decru, Flynn, Wesolowski, Jetchev,
- Signatures:
  - Vercauteren et al., Beullens,…
- Attacks:
  - Petit, Biasse, Bernstein …
- Graph structure:
  - Kohel, Arpin et al.