Quantum attacks against isogenies

Daniel J. Bernstein

1994 Shor discrete-log algorithm:

Input prime p; $g \in \mathbf{F}_p^*$; $h \in g^{\mathbf{Z}}$.

Define $\varphi : \mathbf{Z} \times \mathbf{Z} \to \mathbf{F}_p^*$ by $\varphi(a, b) = g^a h^b$. Fast function.

If $h = g^s$ and g has order N then $\operatorname{Ker} \varphi = \mathbf{Z}(N, 0) + \mathbf{Z}(s, -1)$.

Shor computes φ on quantum superposition of many (a, b); deduces $\operatorname{Ker} \varphi$; deduces s in $\mathbf{Z}/N$.

Shor also generalizes
from $\mathbf{F}_p^*$ to other finite groups
with fast computations.
e.g. $\mathbf{F}_q^*$ for prime power q;
$E(\mathbf{F}_q)$ for elliptic curve $E/\mathbf{F}_q$.

Shor also generalizes
from $\mathbf{F}_p^*$ to other finite groups
with fast computations.
e.g. $\mathbf{F}_q^*$ for prime power q;
$E(\mathbf{F}_q)$ for elliptic curve $E/\mathbf{F}_q$.

1995 Boneh–Lipton:
Find "hidden" lattice $L \subseteq \mathbf{Z}^n$,
given fast function $\varphi : \mathbf{Z}^n \to X$
that induces $\mathbf{Z}^n/L \hookrightarrow X$.

Shor also generalizes from $\mathbf{F}_p^*$ to other finite groups with fast computations.
e.g. $\mathbf{F}_q^*$ for prime power q;
$E(\mathbf{F}_q)$ for elliptic curve $E/\mathbf{F}_q$.

1995 Boneh–Lipton:
Find "hidden" lattice $L \subseteq \mathbf{Z}^n$, given fast function $\varphi : \mathbf{Z}^n \to X$ that induces $\mathbf{Z}^n/L \hookrightarrow X$.

Non-commutative generalizations:
e.g. find hidden subgroup $H \subseteq S_n$, given fast function $\varphi : S_n \to X$ that induces $S_n/H \hookrightarrow X$?
Some progress, some obstacles.

The hidden-shift problem

Given $N \in \mathbf{Z}$, $N > 0$;
$f_0 : \mathbf{Z}/N \hookrightarrow X$; $f_1 : \mathbf{Z}/N \hookrightarrow X$;
$f_1(a) = f_0(a + s)$ for all $a \in \mathbf{Z}/N$.

Goal: Find $s \in \mathbf{Z}/N$.

The hidden-shift problem

Given $N \in \mathbf{Z}$, $N > 0$;
$f_0 : \mathbf{Z}/N \hookrightarrow X$; $f_1 : \mathbf{Z}/N \hookrightarrow X$;
$f_1(a) = f_0(a + s)$ for all $a \in \mathbf{Z}/N$.

Goal: Find $s \in \mathbf{Z}/N$.

Dihedral group $D_N = \mathbf{Z}/N \times \mathbf{Z}/2$:
$(a, b)(c, d) = (a + (-1)^b c, b + d)$.

The hidden-shift problem

Given $N \in \mathbf{Z}$, $N > 0$;
$f_0 : \mathbf{Z}/N \hookrightarrow X$; $f_1 : \mathbf{Z}/N \hookrightarrow X$;
$f_1(a) = f_0(a+s)$ for all $a \in \mathbf{Z}/N$.

Goal: Find $s \in \mathbf{Z}/N$.

Dihedral group $D_N = \mathbf{Z}/N \times \mathbf{Z}/2$:
$(a, b)(c, d) = (a + (-1)^b c, b + d)$.

Define $\varphi : D_N \to X$ by
$\varphi(a, i) = f_i(a)$. Then φ hides
subgroup $\{(0, 0), (s, 1)\}$ of D_N.

The hidden-shift problem

Given $N \in \mathbf{Z}$, $N > 0$;
$f_0 : \mathbf{Z}/N \hookrightarrow X$; $f_1 : \mathbf{Z}/N \hookrightarrow X$;
$f_1(a) = f_0(a + s)$ for all $a \in \mathbf{Z}/N$.

Goal: Find $s \in \mathbf{Z}/N$.

Dihedral group $D_N = \mathbf{Z}/N \times \mathbf{Z}/2$:
$(a, b)(c, d) = (a + (-1)^b c, b + d)$.

Define $\varphi : D_N \to X$ by
$\varphi(a, i) = f_i(a)$. Then φ hides
subgroup $\{(0, 0), (s, 1)\}$ of D_N.

These are the only "Shor-hard"
hidden subgroups of D_N.

1998 Ettinger–Høyer:
Solve hidden-shift problem using $O(\log N)$ quantum φ evaluations, huge φ-independent computation.

1998 Ettinger–Høyer:
Solve hidden-shift problem using $O(\log N)$ quantum φ evaluations, huge φ-independent computation.

(1999–2004 Ettinger–Høyer–Knill: Similarly few evaluations for hidden subgroups of any group.)

1998 Ettinger–Høyer:
Solve hidden-shift problem using $O(\log N)$ quantum φ evaluations, huge φ-independent computation.

(1999–2004 Ettinger–Høyer–Knill: Similarly few evaluations for hidden subgroups of any group.)

2003 Kuperberg:
Solve hidden-shift problem using more quantum φ evaluations, less φ-independent computation.

1998 Ettinger–Høyer:
Solve hidden-shift problem using $O(\log N)$ quantum φ evaluations, huge φ-independent computation.

(1999–2004 Ettinger–Høyer–Knill: Similarly few evaluations for hidden subgroups of any group.)

2003 Kuperberg:
Solve hidden-shift problem using more quantum φ evaluations, less φ-independent computation.

2004 Regev, 2011 Kuperberg:
More tradeoffs, better tradeoffs.

Attacking isogenies

CRS/CSIDH: Class group G acts freely and transitively on a set X of curves over $\mathbf{F}_p$.

Attacking isogenies

CRS/CSIDH: Class group G acts freely and transitively on a set X of curves over $\mathbf{F}_p$.

Usually $G \cong \mathbf{Z}/N$ with $N \approx p^{1/2}$.

Attacking isogenies

CRS/CSIDH: Class group G acts freely and transitively on a set X of curves over $\mathbf{F}_p$.

Usually $G \cong \mathbf{Z}/N$ with $N \approx p^{1/2}$. Compute N by Shor's algorithm.

Attacking isogenies

CRS/CSIDH: Class group G acts freely and transitively on a set X of curves over $\mathbf{F}_p$.

Usually $G \cong \mathbf{Z}/N$ with $N \approx p^{1/2}$. Compute N by Shor's algorithm. Find ideal I with $G = [I]^{\mathbf{Z}}$.

Attacking isogenies

CRS/CSIDH: Class group G acts freely and transitively on a set X of curves over $\mathbf{F}_p$.

Usually $G \cong \mathbf{Z}/N$ with $N \approx p^{1/2}$. Compute N by Shor's algorithm. Find ideal I with $G = [I]^{\mathbf{Z}}$.

Given $E_0, E_1 \in X$: define
$f_0 : \mathbf{Z}/N \hookrightarrow X$ by $a \mapsto [I]^a E_0$;
$f_1 : \mathbf{Z}/N \hookrightarrow X$ by $a \mapsto [I]^a E_1$.

Attacking isogenies

CRS/CSIDH: Class group G acts freely and transitively on a set X of curves over $\mathbf{F}_p$.

Usually $G \cong \mathbf{Z}/N$ with $N \approx p^{1/2}$. Compute N by Shor's algorithm. Find ideal I with $G = [I]^{\mathbf{Z}}$.

Given $E_0, E_1 \in X$: define
$f_0 : \mathbf{Z}/N \hookrightarrow X$ by $a \mapsto [I]^a E_0$;
$f_1 : \mathbf{Z}/N \hookrightarrow X$ by $a \mapsto [I]^a E_1$.

$E_1 = [I]^s E_0$ for some $s \in \mathbf{Z}/N$.

Attacking isogenies

CRS/CSIDH: Class group G acts freely and transitively on a set X of curves over $\mathbf{F}_p$.

Usually $G \cong \mathbf{Z}/N$ with $N \approx p^{1/2}$. Compute N by Shor's algorithm. Find ideal I with $G = [I]^{\mathbf{Z}}$.

Given $E_0, E_1 \in X$: define
$f_0 : \mathbf{Z}/N \hookrightarrow X$ by $a \mapsto [I]^a E_0$;
$f_1 : \mathbf{Z}/N \hookrightarrow X$ by $a \mapsto [I]^a E_1$.

$E_1 = [I]^s E_0$ for some $s \in \mathbf{Z}/N$.
$f_1(a) = f_0(a + s)$ for all $a \in \mathbf{Z}/N$.

Attacking isogenies

CRS/CSIDH: Class group G acts freely and transitively on a set X of curves over $\mathbf{F}_p$.

Usually $G \cong \mathbf{Z}/N$ with $N \approx p^{1/2}$. Compute N by Shor's algorithm. Find ideal I with $G = [I]^{\mathbf{Z}}$.

Given $E_0, E_1 \in X$: define
$f_0 : \mathbf{Z}/N \hookrightarrow X$ by $a \mapsto [I]^a E_0$;
$f_1 : \mathbf{Z}/N \hookrightarrow X$ by $a \mapsto [I]^a E_1$.

$E_1 = [I]^s E_0$ for some $s \in \mathbf{Z}/N$.
$f_1(a) = f_0(a + s)$ for all $a \in \mathbf{Z}/N$.
Find the hidden shift s in f_0, f_1.

How many steps in an action?

Steps for CRS/CSIDH users: fast algorithms for actions of small $[P_1], [P_2], [P_3], \ldots, [P_d]$. e.g., $d = 74$ for CSIDH-512.

How many steps in an action?

Steps for CRS/CSIDH users: fast algorithms for actions of small $[P_1], [P_2], [P_3], \ldots, [P_d]$. e.g., $d = 74$ for CSIDH-512.

$[P_1]^5[P_2]^4[P_3]^1$: 10 steps.

How many steps in an action?

Steps for CRS/CSIDH users:
fast algorithms for actions of
small $[P_1], [P_2], [P_3], \ldots, [P_d]$.
e.g., $d = 74$ for CSIDH-512.

$[P_1]^5[P_2]^4[P_3]^1$: 10 steps.
$[P_1]^{7038304916}$: 7038304916 steps.

How many steps in an action?

Steps for CRS/CSIDH users:
fast algorithms for actions of
small $[P_1], [P_2], [P_3], \ldots, [P_d]$.
e.g., $d = 74$ for CSIDH-512.

$[P_1]^5[P_2]^4[P_3]^1$: 10 steps.
$[P_1]^{7038304916}$: 7038304916 steps.
$[P_1]^a$ for huge $a \in \mathbf{Z}/N$: Hmmm.

How many steps in an action?

Steps for CRS/CSIDH users: fast algorithms for actions of small $[P_1], [P_2], [P_3], \ldots, [P_d]$. e.g., $d = 74$ for CSIDH-512.

$[P_1]^5[P_2]^4[P_3]^1$: 10 steps.
$[P_1]^{7038304916}$: 7038304916 steps.
$[P_1]^a$ for huge $a \in \mathbf{Z}/N$: Hmmm.

Approach 1: Compute lattice $L =$ $\text{Ker}(a_1, \ldots, a_d \mapsto [P_1]^{a_1} \cdots [P_d]^{a_d})$.

How many steps in an action?

Steps for CRS/CSIDH users: fast algorithms for actions of small $[P_1], [P_2], [P_3], \ldots, [P_d]$. e.g., $d = 74$ for CSIDH-512.

$[P_1]^5[P_2]^4[P_3]^1$: 10 steps.
$[P_1]^{7038304916}$: 7038304916 steps.
$[P_1]^a$ for huge $a \in \mathbf{Z}/N$: Hmmm.

Approach 1: Compute lattice $L = \mathrm{Ker}(a_1, \ldots, a_d \mapsto [P_1]^{a_1} \cdots [P_d]^{a_d})$.

Given $a \in \mathbf{Z}^d$, find close $v \in L$: distance $\exp((\log N)^{1/2+o(1)})$ using time $\exp((\log N)^{1/2+o(1)})$.

Approach 2: Increase d up to $\exp((\log N)^{1/2+o(1)})$. Search randomly for small relations.

Approach 2: Increase d up to $\exp((\log N)^{1/2+o(1)})$. Search randomly for small relations.

2010 Childs–Jao–Soukharev:

A. Time $\exp((\log N)^{1/2+o(1)})$ to compute G action by Approach 2.

Approach 2: Increase d up to $\exp((\log N)^{1/2+o(1)})$. Search randomly for small relations.

2010 Childs–Jao–Soukharev:

A. Time $\exp((\log N)^{1/2+o(1)})$ to compute G action by Approach 2.

B. Unfixably flawed argument that Approach 2 beats Approach 1.

Approach 2: Increase d up to $\exp((\log N)^{1/2+o(1)})$. Search randomly for small relations.

2010 Childs–Jao–Soukharev:

A. Time $\exp((\log N)^{1/2+o(1)})$ to compute G action by Approach 2.

B. Unfixably flawed argument that Approach 2 beats Approach 1.

C. Apply Kuperberg (or Regev): Time $\exp((\log N)^{1/2+o(1)})$ to find $g \in G$ with $gE_0 = E_1$.

Approach 2: Increase d up to $\exp((\log N)^{1/2+o(1)})$. Search randomly for small relations.

2010 Childs–Jao–Soukharev:

A. Time $\exp((\log N)^{1/2+o(1)})$ to compute G action by Approach 2.

B. Unfixably flawed argument that Approach 2 beats Approach 1.

C. Apply Kuperberg (or Regev): Time $\exp((\log N)^{1/2+o(1)})$ to find $g \in G$ with $gE_0 = E_1$.

D. Proof assuming only GRH, using provable-factoring ideas.

Approach 3 (mentioned in 2018 Bernstein–Lange–Martindale–Panny): Uniform $(a_1, \ldots, a_d)$ in $\{-c, \ldots, c\}^d$. Choose c somewhat larger than users do.

Not much slowdown in action. Surely $g = [P_1]^{a_1} \cdots [P_d]^{a_d}$ is nearly uniformly distributed in G.

Approach 3 (mentioned in 2018 Bernstein–Lange–Martindale–Panny): Uniform $(a_1, \ldots, a_d)$ in $\{-c, \ldots, c\}^d$. Choose c somewhat larger than users do.

Not much slowdown in action. Surely $g = [P_1]^{a_1} \cdots [P_d]^{a_d}$ is nearly uniformly distributed in G.

Can quickly compute gE_b and image of g in $\mathbf{Z}/N$.

Approach 3 (mentioned in 2018 Bernstein–Lange–Martindale–Panny): Uniform $(a_1, \ldots, a_d)$ in $\{-c, \ldots, c\}^d$. Choose c somewhat larger than users do.

Not much slowdown in action. Surely $g = [P_1]^{a_1} \cdots [P_d]^{a_d}$ is nearly uniformly distributed in G.

Can quickly compute gE_b and image of g in $\mathbf{Z}/N$.

Need more analysis of impact of these redundant representations upon Kuperberg's algorithm.

How fast are the steps?

e.g. CSIDH-512, user distribution on G, error rate $<2^{-32}$ (is this adequate?), nonlinear bit ops:

$\approx 2^{51}$ by 2018 Jao–LeGrow–Leonardi–Ruiz-Lopez.

How fast are the steps?

e.g. CSIDH-512, user distribution on G, error rate $<2^{-32}$ (is this adequate?), nonlinear bit ops:

$\approx 2^{51}$ by 2018 Jao–LeGrow–Leonardi–Ruiz-Lopez.

Many optimizations, detailed analysis: $765325228976 \approx 0.7 \cdot 2^{40}$ by 2018 BLMP Algorithm 8.1.

How fast are the steps?

e.g. CSIDH-512, user distribution on G, error rate $<2^{-32}$ (is this adequate?), nonlinear bit ops:

$\approx 2^{51}$ by 2018 Jao–LeGrow–Leonardi–Ruiz-Lopez.

Many optimizations, detailed analysis: $765325228976 \approx 0.7 \cdot 2^{40}$ by 2018 BLMP Algorithm 8.1.

quantum.isogenies.org: full software and 56-page paper; variations in 512, distrib, 2^{-32}.

How fast are the steps?

e.g. CSIDH-512, user distribution on G, error rate $<2^{-32}$ (is this adequate?), nonlinear bit ops:

$\approx 2^{51}$ by 2018 Jao–LeGrow–Leonardi–Ruiz-Lopez.

Many optimizations, detailed analysis: $765325228976 \approx 0.7 \cdot 2^{40}$ by 2018 BLMP Algorithm 8.1.

`quantum.isogenies.org`: full software and 56-page paper; variations in 512, distrib, 2^{-32}.

Next big challenge: AT analysis.

How many actions + other costs?

2011 Kuperberg estimates "time" $\exp((0.98\ldots + o(1))(\log_2 N)^{1/2})$; compares to 2003 Kuperberg: $\exp((1.23\ldots + o(1))(\log_2 N)^{1/2})$.

How many actions + other costs?

2011 Kuperberg estimates "time"
$\exp((0.98\ldots + o(1))(\log_2 N)^{1/2})$;
compares to 2003 Kuperberg:
$\exp((1.23\ldots + o(1))(\log_2 N)^{1/2})$.

Open: Do better than $1/2$?
Do better than $0.98\ldots$?

How many actions + other costs?

2011 Kuperberg estimates "time"
$\exp((0.98\ldots + o(1))(\log_2 N)^{1/2})$;
compares to 2003 Kuperberg:
$\exp((1.23\ldots + o(1))(\log_2 N)^{1/2})$.

Open: Do better than 1/2?
Do better than $0.98\ldots$?

Exact number of actions? Some work on analysis+optimization:
2003 Kuperberg; 2011 Kuperberg;
2018 Bonnetain–Naya-Plasencia;
2018 Bonnetain–Schrottenloher;
2019 Kuperberg; 2019 Peikert;
2019 Bonnetain–Schrottenloher.