Efficient algorithms for supersingular isogeny Diffie-Hellman

Craig Costello, Patrick Longa, Michael Naehrig

CRYPTO 2016

Microsoft® Research

Forthcoming post-quantum standards…

- Large-scale quantum computers break RSA, finite fields, elliptic curves
- Aug 2015: NSA announces plans to transition to quantum-resistant algorithms
- Aug 2016: NIST announces late 2017 deadline for the algorithms to be considered

National Institute of Standards and Technology

Popular post-quantum public key primitives

• Lattice-based	(e.g., NTRU’98, LWE’05)
• Code-based	(e.g., McEliece’78)
• Hash-based	(e.g., Merkle trees’79)
• Multivariate-based	(e.g., HFE^{v-}’96)
• Isogeny-based	(Jao and De Feo SIDH’11)

Current confidence may be smaller, but so are current key sizes!

WARNING

DO NOT BE DETERRED BY THE WORD SUPERSINGULAR

Isogenies: basic facts

- **Isogeny:** rational map (non-constant) that is a group homomorphism

$$\phi : E_1 \to E_2$$

- Given finite subgroup $G \subset E_1$, there is a unique curve E_2 and isogeny $\phi : E_1 \to E_2$ (up to isomorphism) having kernel G. We write $E_2 = \phi(E_1) = E_1/G$.

- Degree of (separable) isogeny is number of elements in kernel, same as its degree as a rational map

SIDH: history

- **2006 (OIDH)**: Rostovsev and Stolbunov propose ordinary isogeny DH
- **2010 (OIDH break)**: Childs-Jao-Soukharev give quantum subexponential alg.
- **2011 (SIDH)**: Jao and De Feo fix by choosing supersingular curves

Crucial difference: supersingular (i.e., non-ordinary) endomorphism ring is not commutative (resists above attack)

Analogues between Diffie-Hellman instantiations

	DH	ECDH	SIDH
elements	integers g modulo prime	points P in curve group	curves E in isogeny class
secrets	exponents x	scalars k	isogenies ϕ
computations	$g, x \mapsto g^x$	$k, P \mapsto [k]P$	$\phi, E \mapsto \phi(E)$
hard problem	given g, g^x find x	given $P, [k]P$ find k	given $E, \phi(E)$ find ϕ

SIDH in a nutshell:

params public private

$$
\begin{tikzcd}
 & E_0 \arrow[r, Rightarrow, "\phi_A"] \arrow[d, Rightarrow, "\phi_B"'] & E_A = E_0/\langle A\rangle \arrow[d, Rightarrow, "\phi_B'"'] \\
E_0/\langle B\rangle = & E_B \arrow[r, Rightarrow, "{\phi_A}'"] & E_{AB} = E_0/\langle A, B\rangle
\end{tikzcd}
$$

e.g., Alice computes (horizontal) 2-isogenies, Bob computes (vertical) 3-isogenies

SIDH in a nutshell:

params public private

$$\begin{tikzcd}
E_0 \arrow[rr, Rightarrow, "\phi_A"] \arrow[d, Rightarrow, "\phi_B"'] & & E_A = E_0/\langle P_A + [s_A]Q_A \rangle \arrow[d, Rightarrow, "\phi_B'"'] \\
E_0/\langle P_B + [s_B]Q_B \rangle = E_B \arrow[rr, Rightarrow, "\phi_A{}'"] & & E_{AB} = E_0/\langle A, B \rangle
\end{tikzcd}$$

$$(R_A, S_A) = (\phi_A(P_B), \phi_A(Q_B))$$

$$(\phi_B(P_A), \phi_B(Q_A)) = (R_B, S_B)$$

Jao & De Feo's key: Alice sends her isogeny evaluated at Bob's generators, vice versa

$$E_A/\langle R_A + [s_B]S_A \rangle \cong E_0/\langle P_A + [s_A]Q_A\,, P_B + [s_B]Q_B \rangle \cong E_B/\langle R_B + [s_A]S_B \rangle$$

SIDH shared secret is the j-invariant of E_{AB}

SIDH: security

- **Setting:** supersingular elliptic curves $E/\mathbb{F}_{p^2}$ where p is a large prime

- **Hard problem:** Given $P, Q \in E$ and $\phi(P), \phi(Q) \in \phi(E)$, compute ϕ
  (where ϕ has fixed, smooth, public degree)

- **Best (known) attacks:** classical $O(p^{1/4})$ and quantum $O(p^{1/6})$

- **Confidence:** above complexities are optimal for (above generic) claw attack

Exploiting smooth degree isogenies

- Computing isogenies of prime degree ℓ at least $O(\ell)$
- We need exponential #secrets$\leftrightarrow$ #isogenies$\leftrightarrow$#kernel subgroups
- Upshot: isogenies must have exponential degree. Can't compute unless smooth!
- We will only use isogenies of degree ℓ^e for $\ell \in \{2,3\}$

Exploiting smooth degree isogenies

- Suppose secret point R_0 has order 2^{372}, we need $\phi : E \to E/\langle R_0 \rangle$

- Factor $\phi = \phi_{371} \dots \phi_1 \phi_0$, with ϕ_i are 2-isogenies, and walk to $E/\langle R_0 \rangle$

$$
\begin{array}{ll}
\phi_0 \colon E_0 \to E_0/\langle [2^{371}] R_0 \rangle\,, & R_1 = \phi_0(R_0); \\
\phi_1 \colon E_1 \to E_1/\langle [2^{370}] R_1 \rangle\,, & R_2 = \phi_1(R_1); \\
\vdots & \vdots \\
\phi_{370} \colon E_{370} \to E_{370}/\langle [2^{1}] R_{370} \rangle\,, & R_{371} = \phi_{370}(R_{370}); \\
\phi_{371} \colon E_{371} \to E_{371}/\langle R_{371} \rangle\,. &
\end{array}
$$

- The above is naïve: there is a much faster way (see [DJP'14]).
- SIDH requires two types of arithmetic: $[m]P \in E$ and $\phi : E \to E'$

Motivation

Can we actually securely deploy SIDH?

Our performance improvements

1. Projective isogenies → $\mathbb{P}^1$ everywhere
2. Fast $\mathbb{F}_{p^2}$ arithmetic
3. Tight public parameters

(just 1 today...)

Point *and* isogeny arithmetic in $\mathbb{P}^1$

ECDH: move around different points on a fixed curve.

SIDH: move around different points and different curves

$$\boxed{E_{\mathrm{a,b}} : \quad by^2 = x^3 + ax^2 + x}$$

$(x, y) \leftrightarrow (X : Y : Z)$ $\quad\Big\downarrow\quad$ $(a, b) \leftrightarrow (A : B : C)$

$$\boxed{E_{(\mathrm{A:B:C})} : \quad BY^2Z = CX^3 + AX^2Z + CXZ^2}$$

The Montgomery B coefficient only fixes the quadratic twist. Can ignore it in SIDH since $j(E) = j(E')$

$\mathbb{P}^1$ point arithmetic (Montgomery): $(X : Z) \mapsto (X' : Z')$

$\mathbb{P}^1$ isogeny arithmetic (this work): $(A : C) \mapsto (A' : C')$

Parameters

params public private

$$p = 2^{372}3^{239} - 1$$

$p \approx 2^{768}$ gives $\approx$ 192 bits classical and 128 bits quantum security against best known attacks

$$E_0 / \mathbb{F}_{p^2} : y^2 = x^3 + x$$

$$\#E_0 = (p+1)^2 = (2^{372}3^{239})^2$$

Easy ECDLP

$$P_A, P_B \in E_0(\mathbb{F}_p),\ Q_A = \tau(P_A),\ Q_B = \tau(P_B)$$

376 bytes

48 bytes

$$s_A, s_B \in \mathbb{Z}$$

$$\mathrm{PK} = [x(P), x(Q), x(Q-P)] \in \left(\mathbb{F}_{p^2}\right)^3$$

564 bytes

188 bytes

$$j(E_{AB}) \in \mathbb{F}_{p^2}$$

Performance benchmarks

SIDH operation	This work*	Prior work (AFJ'14)
Alice key generation	46	149
Bob key generation	52	152
Alice shared secret	44	118
Bob shared secret	50	122
Total	**192**	**540**

Table: millions of clock cycles for DH operations on 3.4GHz Intel Core i7-4770 (Haswell)

*includes full protection against timing and cache attacks

BigMont: a strong SIDH+ECDH hybrid

- No clear frontrunner for PQ key exchange
- Hybrid particularly good idea for (relatively young) SIDH
- Hybrid particularly easy for SIDH

There are exponentially many A such that $E_A\,/\mathbb{F}_{p^2}\colon y^2 = x^3 + Ax^2 + x$ is in the supersingular isogeny class. These are all unsuitable for ECDH.

There are also exponentially many A such that $E_A\,/\mathbb{F}_{p^2}\colon y^2 = x^3 + Ax^2 + x$ is suitable for ECDH, e.g. $A = 624450$.

SIDH vs. SIDH+ECDH hybrid

comparison		SIDH	SIDH+ECDH
bit security (hard problem)	classical	192 (SSDDH)	384 (ECDHP)
	quantum	128 (SSDDH)	128 (SSDDH)
public key size (bytes)		564	658
Speed (cc x 10^6)	Alice key gen.	46	52
	Bob key gen.	52	58
	Alice shared sec.	44	50
	Bob shared sec.	50	57

Colossal amount of classical security almost-for-free (≈ no more code)

SIDH vs. lattice "DH" primitives

Name	Primitive	Full DH (ms)	PK size (bytes)
Frodo	LWE	2.600	11,300
NewHope	R-LWE	0.310	1,792
NTRU	NTRU	2.429	1,024
SIDH	Supersingular Isogeny	900	564

Table: ms for full DH round (Alice + Bob) on 2.6GHz Intel Xeon i5 (Sandy Bridge)
See "Frodo" for benchmarking details.

All numbers above are for plain C implementations (e.g., SIDH w. assembly optimizations is 56ms)

Validating public keys

- Issues regarding public key validation: Asiacrypt2016 paper by Galbraith-Petit-Shani-Ti

- NSA countermeasure: “Failure is not an option: standardization issues for PQ key agreement”

- Thus, library currently supports ephemeral DH only

Thanks!

Full version

http://eprint.iacr.org/2016/413

SIDH library

https://www.microsoft.com/en-us/research/project/sidh-library/