SIKE (in Round 2)

Reza Azarderakhsh, Matthew Campagna, Craig Costello, Luca De Feo, Basil Hess, David Jao, Brian Koziel, Geovandro Pereira, Brian LaMacchia, Patrick Longa, Michael Naehrig, Joost Renes, Vladimir Soukharev
SIKE Round 2 updates

- **Smaller parameters**: attacks are worse in practice
- **Compression**: even smaller public keys / ciphertexts
- **New starting curve**: *a bit* better
ECC vs. post-quantum ECC
Alice $2^e$-isogenies, Bob $3^f$-isogenies

Diffie-Hellman instantiations

<table>
<thead>
<tr>
<th>Elements</th>
<th>DH</th>
<th>ECDH</th>
<th>SIDH/SIKE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>integers $g$ modulo prime</td>
<td>points $P$ in curve group</td>
<td>curves $E$ in isogeny class</td>
</tr>
<tr>
<td>Secrets</td>
<td>exponents $x$</td>
<td>scalars $k$</td>
<td>isogenies $\phi$</td>
</tr>
<tr>
<td>computations</td>
<td>$g, x \mapsto g^x$</td>
<td>$k, P \mapsto [k]P$</td>
<td>$\phi, E \mapsto \phi(E)$</td>
</tr>
<tr>
<td>hard problem</td>
<td>given $g, g^x$</td>
<td>given $P, [k]P$</td>
<td>given $E, \phi(E)$</td>
</tr>
</tbody>
</table>
SIDH/SIKE setup

\[ p = 2^i \cdot 3^j - 1 \]

• Elements are supersingular elliptic curves over \( \mathbb{F}_{p^2} \) (up to \( \cong \))

• Roughly \( p/12 \) of them

• For any \( \ell \) (not a multiple of \( p \)), set forms a \((\ell + 1)\)-regular graph that is Ramanujan: edges are isogenies, \( \ell \in \{2, 3\} \) means they’re \( \mathbb{F}_{p^2} \)-rational

• Easiest with an example...
Supersingular isogeny graph for $\ell = 2$: $X(S_{241^2}, 2)$
Supersingular isogeny graph for $\ell = 3$: $X(S_{241^2}, 3)$
**Cyclic subgroup isogenies**

- Maps $\phi : E \to E'$ that are (algebraic/geometric) morphisms $(x, y) \mapsto (x', y')$
- Similar to (e.g.) multiplication-by-$n$, except we land on a different curve

$E[n] \cong \mathbb{Z}_n \times \mathbb{Z}_n$

- Kernel of $[n] \cong \mathbb{Z}_n \times \mathbb{Z}_n$
  - Degree is $n^2$
- Kernel of cyclic $n$-isogeny $\cong \mathbb{Z}_n$
  - Degree is $n$
E.g. Montgomery 2-isogeny

\[ E : \quad y^2 = x^3 + Ax^2 + x \quad \quad E' : \quad y^2 = x^3 + A'x^2 + x \]

\[ E[2] = \{ O_E, (0,0), (\alpha,0), (1/\alpha, 0) \} \]

\[ [2] : E \rightarrow E, \quad x \mapsto \frac{(x^2 - 1)^2}{4x(x^2 + Ax + x)} \quad \ker([2]) = E[2] \]

\[ \phi : E \rightarrow E', \quad x \mapsto x \cdot \left( \frac{\alpha x - 1}{x - \alpha} \right) \quad \ker(\phi) = \{ O_E, (\alpha,0) \} \]

In practice we work entirely in \( \mathbb{P}^1 \), i.e., \((X:Z) \mapsto (X':Z')\), etc.
Computing $\ell^e$ degree isogenies

(suppose $\ell = 2$ and $e = 6$)

$\phi : E_0 \to E_6$ is degree 64

64 elements in its kernel

$\ker(\phi) = \langle P_0 \rangle$

$E_6 = E_0 / \langle P_0 \rangle$
Computing $\ell^e$ degree isogenies

(suppose $\ell = 2$ and $e = 6$)

$\phi : E_0 \to E_6$ is degree 64

64 elements in its kernel

$\ker(\phi) = \langle P_0 \rangle$

$E_5 = E_0/\langle [2]P_0 \rangle$
Computing $\ell^e$ degree isogenies

(suppose $\ell = 2$ and $e = 6$)

$\phi : E_0 \rightarrow E_6$ is degree 64
64 elements in its kernel
$\ker(\phi) = \langle P_0 \rangle$

$E_4 = E_0 / \langle [4]P_0 \rangle$
Computing $\ell^e$ degree isogenies

(suppose $\ell = 2$ and $e = 6$)

$\phi : E_0 \to E_6$ is degree 64
64 elements in its kernel
$\ker(\phi) = \langle P_0 \rangle$

$E_3 = E_0 / \langle [8]P_0 \rangle$
Computing $\ell^e$ degree isogenies

(suppose $\ell = 2$ and $e = 6$)

$\phi : E_0 \to E_6$ is degree 64
64 elements in its kernel
$\ker(\phi) = \langle P_0 \rangle$

$E_2 = E_0/\langle [16]P_0 \rangle$
Computing $\ell^e$ degree isogenies
(suppose $\ell = 2$ and $e = 6$)

$\phi : E_0 \to E_6$ is degree 64
64 elements in its kernel
$\ker(\phi) = \langle P_0 \rangle$

$E_1 = E_0 / \langle [32]P_0 \rangle = \phi_0(E_0)$
Computing $\ell^e$ degree isogenies

(suppose $\ell = 2$ and $e = 6$)

$\phi : E_0 \rightarrow E_6$ is degree 64

64 elements in its kernel

$\ker(\phi) = \langle P_0 \rangle$

$E_1 = E_0/\langle [32]P_0 \rangle = \phi_0(E_0)$

$P_1 = \phi_0(P_0)$
Computing $\ell^e$ degree isogenies

(suppose $\ell = 2$ and $e = 6$)

$\phi : E_0 \to E_6$ is degree 64
64 elements in its kernel
$\ker(\phi) = \langle P_0 \rangle$

$E_6 = E_1/\langle P_1 \rangle$
Computing $\ell^e$ degree isogenies

(suppose $\ell = 2$ and $e = 6$)

$\phi : E_0 \to E_6$ is degree 64

64 elements in its kernel

$\ker(\phi) = \langle P_0 \rangle$

$E_5 = E_1/\langle [2]P_1 \rangle$
Computing $\ell^e$ degree isogenies

(suppose $\ell = 2$ and $e = 6$)

$\phi : E_0 \to E_6$ is degree 64

64 elements in its kernel

$\text{ker}(\phi) = \langle P_0 \rangle$

$E_4 = E_1/\langle [4]P_1 \rangle$
Computing $\ell^e$ degree isogenies

(suppose $\ell = 2$ and $e = 6$)

$\phi : E_0 \rightarrow E_6$ is degree 64
64 elements in its kernel
$\ker(\phi) = \langle P_0 \rangle$

$E_3 = E_1/\langle [8]P_1 \rangle$
Computing $\ell^e$ degree isogenies

(suppose $\ell = 2$ and $e = 6$)

$\phi : E_0 \to E_6$ is degree 64

64 elements in its kernel

$\ker(\phi) = \langle P_0 \rangle$

$E_2 = E_1/\langle [16]P_1 \rangle = \phi_1(E_1)$
Computing $\ell^e$ degree isogenies

(suppose $\ell = 2$ and $e = 6$)

$\phi : E_0 \to E_6$ is degree 64

64 elements in its kernel

$\ker(\phi) = \langle P_0 \rangle$

$E_2 = E_1/\langle [16]P_1 \rangle = \phi_1(E_1)$

$P_2 = \phi_1(P_1)$
Computing $\ell^e$ degree isogenies

(suppose $\ell = 2$ and $e = 6$)

$\phi : E_0 \to E_6$ is degree 64

64 elements in its kernel

$\ker(\phi) = \langle P_0 \rangle$

$E_6 = E_2 / \langle P_2 \rangle$
Computing $\ell^e$ degree isogenies

(suppose $\ell = 2$ and $e = 6$)

$\phi : E_0 \to E_6$ is degree 64
64 elements in its kernel
$\ker(\phi) = \langle P_0 \rangle$

$E_5 = E_2 / \langle [2]P_2 \rangle$
Computing $\ell^e$ degree isogenies

(suppose $\ell = 2$ and $e = 6$)

$\phi : E_0 \to E_6$ is degree 64

64 elements in its kernel

$\ker(\phi) = \langle P_0 \rangle$

$E_4 = E_2/\langle [4]P_2 \rangle$
Computing $\ell^e$ degree isogenies

(suppose $\ell = 2$ and $e = 6$)

$\phi : E_0 \to E_6$ is degree 64

64 elements in its kernel

$\ker(\phi) = \langle P_0 \rangle$

$E_3 = E_2/\langle [8]P_2 \rangle$

$= \phi_2(E_2)$
Computing $\ell^e$ degree isogenies

(suppose $\ell = 2$ and $e = 6$)

$\phi : E_0 \to E_6$ is degree 64

64 elements in its kernel

$\ker(\phi) = \langle P_0 \rangle$

$E_3 = E_2/\langle [8]P_2 \rangle = \phi_2(E_2)$

$P_3 = \phi_2(P_2)$
Computing $\ell^e$ degree isogenies

(suppose $\ell = 2$ and $e = 6$)

$\phi : E_0 \to E_6$ is degree 64

64 elements in its kernel

$\ker(\phi) = \langle P_0 \rangle$

$E_6 = E_3/\langle P_3 \rangle$
Computing $\ell^e$ degree isogenies

(suppose $\ell = 2$ and $e = 6$)

$\phi : E_0 \to E_6$ is degree 64

64 elements in its kernel

$\ker(\phi) = \langle P_0 \rangle$

$E_5 = E_3/\langle [2]P_3 \rangle$
Computing $\ell^e$ degree isogenies

(suppose $\ell = 2$ and $e = 6$)

$\phi : E_0 \to E_6$ is degree 64
64 elements in its kernel
$\ker(\phi) = \langle P_0 \rangle$

$E_4 = E_3/\langle [4]P_3 \rangle$
Computing $\ell^e$ degree isogenies

(suppose $\ell = 2$ and $e = 6$)

$\phi : E_0 \rightarrow E_6$ is degree 64
64 elements in its kernel

$\ker(\phi) = \langle P_0 \rangle$

$E_4 = E_3/\langle [4]P_3 \rangle$

$P_4 = \phi_3(P_3)$
Computing $\ell^e$ degree isogenies

(suppose $\ell = 2$ and $e = 6$)

$\phi : E_0 \to E_6$ is degree 64

64 elements in its kernel

$\ker(\phi) = \langle P_0 \rangle$

$E_5 = E_4/\langle[2]P_4\rangle$
Computing $\ell^e$ degree isogenies

(suppose $\ell = 2$ and $e = 6$)

$\phi : E_0 \rightarrow E_6$ is degree 64
64 elements in its kernel
$\ker(\phi) = \langle P_0 \rangle$

$E_5 = E_4/\langle [2]P_4 \rangle$

$P_5 = \phi_4(P_4)$
Computing $\ell^e$ degree isogenies

(suppose $\ell = 2$ and $e = 6$)

$\phi : E_0 \to E_6$ is degree 64

64 elements in its kernel

$\ker(\phi) = \langle P_0 \rangle$

$E_6 = E_5/\langle P_5 \rangle$
Optimal strategies
Optimal strategies

\[ n^2 \rightarrow n \log n \]
Computing $\ell^e$ degree isogenies

$\phi : E_0 \to E_6$

$\phi = \phi_5 \circ \phi_4 \circ \phi_3 \circ \phi_2 \circ \phi_1 \circ \phi_0$
Rest of talk: given $E, E'$, find path (of known length)...

\[ E \quad ? \quad E' \]
Claw algorithm: meet-in-the-middle

Given $E$ and $E' = \phi(E)$, with $\phi$ degree $\ell^e$, find $\phi$
Claw algorithm: meet-in-the-middle

Compute and store $\ell^{e/2}$-isogenies on one side
Claw algorithm: meet-in-the-middle

Compute and store $\ell^{e/2}$-isogenies on one side
Claw algorithm: meet-in-the-middle

... until you have all of them
Now compute $\ell^{e/2}$-isogenies on the other side
Claw algorithm: meet-in-the-middle

... discarding them until you find a collision
Claw algorithm: meet-in-the-middle

... discarding them until you find a collision
Claw algorithm: meet-in-the-middle

... discarding them until you find a collision
Collision will most likely be unique shortest path
Claw algorithm: meet-in-the-middle

This path describes secret isogeny \( \phi : E \to E' \)
Claw algorithm: classical analysis

• There are $O(\ell^{e/2})$ curves $\ell^{e/2}$-isogenous to $E'$ (the blue nodes).

  thus $O(\ell^{e/2}) = O(p^{1/4})$ classical memory

• There are $O(\ell^{e/2})$ curves $\ell^{e/2}$-isogenous to $E'$ (the blue nodes), and there are $O(\ell^{e/2})$ curves $\ell^{e/2}$-isogenous to $E$ (the purple nodes).

  thus $O(\ell^{e/2}) = O(p^{1/4})$ classical time

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

• Confidence: both complexities are optimal for a black-box claw attack
The curves and their security estimates

\[ p = 2^{e_A}3^{e_B} - 1 \]

<table>
<thead>
<tr>
<th>Target Security Level</th>
<th>Name</th>
<th>((e_A, e_B))</th>
<th>(k)</th>
<th>(2^{k-1})</th>
<th>(\min (\sqrt{2^{e_A}}, \sqrt{3^{e_B}}))</th>
<th>(\sqrt{2}^k)</th>
<th>(\min (\sqrt[3]{2^{e_A}}, \sqrt[3]{3^{e_B}}))</th>
</tr>
</thead>
<tbody>
<tr>
<td>NIST 1</td>
<td>SIKEp503</td>
<td>(250,159)</td>
<td>128</td>
<td>2^{127}</td>
<td>2^{125}</td>
<td>2^{64}</td>
<td>2^{83}</td>
</tr>
<tr>
<td>NIST 3</td>
<td>SIKEp761</td>
<td>(372,239)</td>
<td>192</td>
<td>2^{191}</td>
<td>2^{186}</td>
<td>2^{96}</td>
<td>2^{124}</td>
</tr>
<tr>
<td>NIST 5</td>
<td>SIKEp964</td>
<td>(486,301)</td>
<td>256</td>
<td>2^{255}</td>
<td>2^{238}</td>
<td>2^{128}</td>
<td>2^{159}</td>
</tr>
</tbody>
</table>

\(e\) classically
\(\sqrt{\omega}\) quantum
Since submission...

**cryptanalysis**

- Adj, Cervantes-Vázquez, Chi-Domínguez, Menezes, Rodríguez-Henríquez: *On the cost of computing isogenies between supersingular elliptic curves* (ia.cr/2018/313)

- Jaques-Schanck: *Quantum cryptanalysis in the RAM model: claw-finding attacks on SIKE* (ia.cr/2019/103)

- C-Longa-Naehrig-Renes-Virdia: *Improved classical cryptanalysis of the computational supersingular isogeny problem* (ia.cr/2019/XXX)

**compression**

- Zanon, Simplicio Jr, Pereira, Doliskani, Barreto: *Faster key compression for isogeny-based cryptosystems* (ia.cr/2017/1143)
Jaques-Schanck (ia.cr/2019/103)

• Models allow direct classical-quantum comparison: best known quantum algorithms do not achieve significant advantage over classical

• (w.r.t. Tani and Grover) In certain attack scenarios classical security is the limiting factor for achieving a specified security level

• “Our conclusion is that an adversary with enough memory to run Tani's algorithm with the query-optimal parameters could break SIKE faster by using the classical control hardware to run vOW"
van Oorschot-Wiener

Do not have enough memory to MitM, so run a deterministic function that combines both sides into a set $S$.

$$f_n : S \rightarrow S$$

$$x_i \mapsto x_{i+1}$$

$f_n :$ a half-sized isogeny + $\epsilon$
$E_0$
can’t possibly store all these: fix $w$ as upper bound on $\#x_i$ storage

store fraction $0 < \theta \ll 1$
vOW

- $f_n$ is a deterministic random function, different for each $IV = n$

- For a fixed $n$, each processor does the following:
  
  - pick a random starting point $x_0$
  - produce trail $x_i = f_n(x_{i-1})$, for $i = 1, 2, ...$
  - stop when $x_d$ is “distinguished” ($1/\theta$).

    if ($x_d$ has not been seen yet) then
      store triple $(x_0, x_d, d)$ and resample
    else
      if (collision not “golden”) then
        overwrite previous triple $(x_0, x_d, d)$ and resample
      else
        ...
Trails and collisions

how should we set $\theta$?

some will be longer than $1/\theta$

some will be shorter than $1/\theta$

how long’s too long?

how do we check collisions?

and what does check mean?
Checking collisions

memory
($x_0, x_d, d$)
($x'_0, x'_e, e$)
Checking collisions

memory
$(x_0, x_d, d)$
$(x'_0, x'_e, e)$

$x_0 \leftarrow f(x'_0)$
Checking collisions

memory
$(x_0, x_d, d)$
$(x'_0, x'_e, e)$

$x_0 \leftarrow f(x'_0)$
Checking collisions

memory
$(x_0, x_d, d)$
$(x'_0, x'_e, e)$

$f_n(x_0) \neq f_n(x'_0)$
Checking collisions

memory
$(x_0, x_d, d)$
$(x'_e, d, e)$

$f_n(x_0) \neq f_n(x'_0)$
Checking collisions

memory
\((x_0, x_d, d)\)
\((x'_0, x'_e, e)\)

\(f_n(x_0) \neq f_n(x'_0)\)
Checking collisions

memory
$(x_0, x_d, d)$
$(x'_0, x'_e, e)$

$f_n(x_0) \neq f_n(x'_0)$
Checking collisions

memory

\((x_0, x_d, d)\)

\((x'_0, x'_e, e)\)

\[ f_n(x_0) = f_n(x'_0) \]

\[ x_0 \neq x'_0 \]

DONE?
Checking collisions

\[ f_n(x_0) = f_n(x'_0) \]

memory \((x_0, x_d, d')\), \((x'_0, x'_e, e)\)

Nope! False alarm
Random collisions vs. the golden collision

• A random function \( f_n : S \rightarrow S \) has many collisions, e.g., think of the random function as a hash function (it kinda is anyway)

• We will encounter many of these before we hit the one we want, i.e., the “golden collision”

• Much of the algorithm is spent walking, much is spent checking useless annoying collisions

• Ideally there’ll be many paths that take us to the golden collision...
Random $f_n$: the good, the bad and the ugly...

- Even more annoying is that we have to restart the whole algorithm, time and time again...
vOW Complexity

- Analysis conducted by van Oorschot and Wiener
- Analysis confirmed (for CSSI) by Adj et al.
- Analysis re-confirmed (for CSSI) by Jaques-Schanck
- Analysis re-re-confirmed (for CSSI) by us

\[ T \approx 2.5 \sqrt{N^3/w \cdot t} \]

- \( T \) = time taken to find golden collision
- \( N = |S| \), the number of \( x_i \), approx. \( p^{1/4} \)
- \( w \) = the maximum number of \( x_i \) that can be stored.
- \( t \) = the time taken to compute \( f_n: x_i \mapsto x_{i+1} \) (i.e., half-sized isogeny+\( \epsilon \))
vOW security \((w = 2^{80})\)

<table>
<thead>
<tr>
<th>NIST level</th>
<th>Name</th>
<th>((e_A, e_B))</th>
<th>(\log_2(N))</th>
<th>(\log_2(\text{vOW}))</th>
<th>(\log_2(N))</th>
<th>(\log_2(\text{vOW}))</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>SIKEp434</td>
<td>(216, 137)</td>
<td>107</td>
<td>143</td>
<td>107</td>
<td>144</td>
</tr>
<tr>
<td>3</td>
<td>SIKEp610</td>
<td>(305, 192)</td>
<td>151</td>
<td>210</td>
<td>150</td>
<td>210</td>
</tr>
<tr>
<td>5</td>
<td>SIKEp751</td>
<td>(372, 239)</td>
<td>185</td>
<td>262</td>
<td>188</td>
<td>268</td>
</tr>
</tbody>
</table>

\(\log_2(\text{vOW})\): count of number of x64 instructions required to mount vOW. Intended as conservative lower-bound on the classical gate count.
## Uncompressed SIKE

<table>
<thead>
<tr>
<th>NIST level</th>
<th>Round 1</th>
<th></th>
<th></th>
<th>Round 2</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>prime (bits)</td>
<td>PK size (bytes)</td>
<td>cycles (m) (enc+dec)</td>
<td></td>
<td>prime (bits)</td>
</tr>
<tr>
<td>1</td>
<td>503</td>
<td>378</td>
<td>30.7</td>
<td>434</td>
<td>326</td>
<td>21.9</td>
</tr>
<tr>
<td>3</td>
<td>751</td>
<td>564</td>
<td>88.5</td>
<td>610</td>
<td>458</td>
<td>52.8</td>
</tr>
<tr>
<td>5</td>
<td>964</td>
<td>723</td>
<td>-</td>
<td>751</td>
<td>564</td>
<td>88.5</td>
</tr>
<tr>
<td>Sec. (NIST)</td>
<td>SIKE prime (bits)</td>
<td>uncompressed PK size (bytes)</td>
<td>Cycles (m) (enc+dec)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>------------</td>
<td>-------------------</td>
<td>-----------------------------</td>
<td>----------------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>434</td>
<td>326</td>
<td>21.9</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>610</td>
<td>458</td>
<td>52.8</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>751</td>
<td>564</td>
<td>88.5</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>compressed PK size (bytes)</th>
<th>Cycles (m) (enc+dec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>191</td>
<td>tbd.</td>
</tr>
<tr>
<td>268</td>
<td>tbd.</td>
</tr>
<tr>
<td>330</td>
<td>tbd.</td>
</tr>
</tbody>
</table>
questions?