AN EXPOSURE MODEL FOR SUPERSINGULAR ISOGENY DIFFIE-HELLMAN KEY EXCHANGE

Brian Koziel, Reza Azarderakhsh, David Jao

Assistant Professor
Florida Atlantic University
Current PKC is safe until large-scale quantum computers are available

- **ECDH, ECDSA**: Protected by the **Elliptic curve discrete logarithm** problem

---

*Figure: Quantum Computer (The Verge)*
Current PKC is safe until large-scale quantum computers are available

- **ECDH, ECDSA**: Protected by the Elliptic curve discrete logarithm problem
- **RSA**: Protected by the factorization and discrete logarithm problems
Current PKC is safe until large-scale quantum computers are available

- **ECDH, ECDSA**: Protected by the **Elliptic curve discrete logarithm problem**
- **RSA**: Protected by the **factorization and discrete logarithm problems**
- Large-scale quantum computers with Shor’s algorithm will **BREAK** the security assumptions for these primitives

*Figure: Quantum Computer (The Verge)*
NIST has started a PQC standardization process

Primary Post-Quantum Cryptography (PQC) Candidates:

- Code-Based: McEliece
NIST has started a PQC standardization process

Primary Post-Quantum Cryptography (PQC) Candidates:

- Code-Based: McEliece
- Hash-based: Lamport, Merkle Signatures
NIST has started a PQC standardization process

Primary Post-Quantum Cryptography (PQC) Candidates:

- Code-Based: McEliece
- Hash-based: Lamport, Merkle Signatures
- Lattice-based: NTRU, LWE
NIST has started a PQC standardization process

Primary Post-Quantum Cryptography (PQC) Candidates:

- Code-Based: McEliece
- Hash-based: Lamport, Merkle Signatures
- Lattice-based: NTRU, LWE
- Multivariate: Rainbow Signature
NIST has started a PQC standardization process

Primary Post-Quantum Cryptography (PQC) Candidates:

- Code-Based: McEliece
- Hash-based: Lamport, Merkle Signatures
- Lattice-based: NTRU, LWE
- Multivariate: Rainbow Signature
- Isogeny-based: SIDH, SIKE
NIST has started a PQC standardization process

Primary Post-Quantum Cryptography (PQC) Candidates:

- **Isogeny-based:** SIDH, SIKE

Figure: $E : y^2 = x^3 - x$ Left: $E/\mathbb{Z}$ Right: $E/\mathbb{F}_{127}$
SIDH offers the smallest key sizes of known quantum-resistant algorithms.

Table: Comparison of different post-quantum key exchange and encryption algorithms at 128-bit quantum security level. Key sizes are in Bytes.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>NTRU</th>
<th>New Hope</th>
<th>McBits</th>
<th>SIDH</th>
<th>Compressed SIDH</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Lattice</td>
<td>Ring-LWE</td>
<td>Code</td>
<td>Isogeny</td>
<td>Isogeny</td>
</tr>
<tr>
<td>Public Key</td>
<td>6,130</td>
<td>2,048</td>
<td>1,046,739</td>
<td>576</td>
<td>336</td>
</tr>
<tr>
<td>Private Key</td>
<td>6,743</td>
<td>2,048</td>
<td>10,992</td>
<td>48</td>
<td>48</td>
</tr>
<tr>
<td>Performance</td>
<td>Slow</td>
<td>Very Fast</td>
<td>Slow</td>
<td>Very Slow</td>
<td>Very Slow</td>
</tr>
</tbody>
</table>

Small key sizes reduce transmission cost and storage requirement.
SIDH offers the smallest key sizes of known quantum-resistant algorithms

Table: Comparison of different post-quantum key exchange and encryption algorithms at 128-bit quantum security level. Key sizes are in Bytes

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>NTRU</th>
<th>New Hope</th>
<th>McBits</th>
<th>SIDH</th>
<th>Compressed SIDH</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Lattice</td>
<td>Ring-LWE</td>
<td>Code</td>
<td>Isogeny</td>
<td>Isogeny</td>
</tr>
<tr>
<td>Public Key</td>
<td>6,130</td>
<td>2,048</td>
<td>1,046,739</td>
<td>576</td>
<td>336</td>
</tr>
<tr>
<td>Private Key</td>
<td>6,743</td>
<td>2,048</td>
<td>10,992</td>
<td>48</td>
<td>48</td>
</tr>
<tr>
<td>Performance</td>
<td>Slow</td>
<td>Very Fast</td>
<td>Slow</td>
<td>Very Slow</td>
<td>Very Slow</td>
</tr>
</tbody>
</table>

- Small key sizes reduce transmission cost and storage requirement
Why would I want to use SIDH or SIKE as a quantum alternative?

Pros :)  
- Very small public/private keys  
- Implementations resemble ECC  
- Security based on supersingular isogeny problem  
- **SIKE:** IND-CCA KEM alternative to SIDH → static keys can be reused!  
- No possibility for decryption error  
- No complicated error distributions, rejection sampling, etc.  
- Conservative security analysis when assuming generic attacks

Cons :(  
- Newest candidate for PQC applications  
- Very slow  
- SIDH has security concerns if keys are reused
Isogeny-Based Cryptography History

- 1996: Couveignes - First mention of isogenies in cryptography. Published in 2006
Isogeny-Based Cryptography History

- 1996: Couveignes - First mention of isogenies in cryptography. Published in 2006
- 1999: Galbraith - First published cryptanalysis of isogeny problem
Isogeny-Based Cryptography History

- 1996: Couveignes - First mention of isogenies in cryptography. Published in 2006
- 1999: Galbraith - First published cryptanalysis of isogeny problem
- 2009: Charles et al. - Hash functions from supersingular isogenies
Isogeny-Based Cryptography History

- 1996: Couveignes - First mention of isogenies in cryptography. Published in 2006
- 1999: Galbraith - First published cryptanalysis of isogeny problem
- 2009: Charles et al. - Hash functions from supersingular isogenies
- 2010: Stolbunov - First published isogeny-based public-key cryptosystem based on isogenies between ordinary curves
Isogeny-Based Cryptography History

- 1996: Couveignes - First mention of isogenies in cryptography. Published in 2006
- 1999: Galbraith - First published cryptanalysis of isogeny problem
- 2009: Charles et al. - Hash functions from supersingular isogenies
- 2010: Stolbunov - First published isogeny-based public-key cryptosystem based on isogenies between ordinary curves
- 2010: Childs et al. - Quantum subexponential attack on Stolbunov’s public-key cryptosystem
Isogeny-Based Cryptography History

- 1996: Couveignes - First mention of isogenies in cryptography. Published in 2006
- 1999: Galbraith - First published cryptanalysis of isogeny problem
- 2009: Charles et al. - Hash functions from supersingular isogenies
- 2010: Stolbunov - First published isogeny-based public-key cryptosystem based on isogenies between ordinary curves
- 2010: Childs et al. - Quantum subexponential attack on Stolbunov’s public-key cryptosystem
- 2011: Jao and De Feo - Supersingular Isogeny Diffie-Hellman (SIDH) proposed
Isogeny-Based Cryptography History

- 1996: Couveignes - First mention of isogenies in cryptography. Published in 2006
- 1999: Galbraith - First published cryptanalysis of isogeny problem
- 2009: Charles et al. - Hash functions from supersingular isogenies
- 2010: Stolbunov - First published isogeny-based public-key cryptosystem based on isogenies between ordinary curves
- 2010: Childs et al. - Quantum subexponential attack on Stolbunov’s public-key cryptosystem
- 2011: Jao and De Feo - **Supersingular Isogeny Diffie-Hellman** (SIDH) proposed
- 2016: Galbraith et al. - Active attack against SIDH with static key re-use
Isogeny-Based Cryptography History

- 1996: Couveignes - First mention of isogenies in cryptography. Published in 2006
- 1999: Galbraith - First published cryptanalysis of isogeny problem
- 2009: Charles et al. - Hash functions from supersingular isogenies
- 2010: Stolbunov - First published isogeny-based public-key cryptosystem based on isogenies between ordinary curves
- 2010: Childs et al. - Quantum subexponential attack on Stolbunov’s public-key cryptosystem
- 2011: Jao and De Feo - Supersingular Isogeny Diffie-Hellman (SIDH) proposed
- 2016: Galbraith et al. - Active attack against SIDH with static key re-use
- 2017: Jao et al. - Supersingular Isogeny Key Encapsulation (SIKE) submitted to NIST PQC process
Isogeny-Based Cryptography underlying security

Consider two supersingular elliptic curves defined over a large prime extension field
Isogeny-Based Cryptography underlying security

Consider two supersingular elliptic curves defined over a large prime extension field

- $E_1/\mathbb{F}_{p^2}$ and $E_2/\mathbb{F}_{p^2}$, where $p$ is a large prime
Consider two supersingular elliptic curves defined over a large prime extension field

- $E_1/\mathbb{F}_{p^2}$ and $E_2/\mathbb{F}_{p^2}$, where $p$ is a large prime
- There exists some isogeny $\phi : E_1 \rightarrow E_2$ with a fixed, smooth degree that is public
Consider two supersingular elliptic curves defined over a large prime extension field

- $E_1 / \mathbb{F}_{p^2}$ and $E_2 / \mathbb{F}_{p^2}$, where $p$ is a large prime
- There exists some isogeny $\phi : E_1 \to E_2$ with a fixed, smooth degree that is public

**Supersingular Isogeny Problem**

Given $P, Q \in E_1$ and $\phi(P), \phi(Q) \in E_2$, compute the secret isogeny, $\phi$
Consider two supersingular elliptic curves defined over a large prime extension field

- \( E_1 / \mathbb{F}_{p^2} \) and \( E_2 / \mathbb{F}_{p^2} \), where \( p \) is a large prime
- There exists some isogeny \( \phi : E_1 \rightarrow E_2 \) with a fixed, smooth degree that is public

**Supersingular Isogeny Problem**

Given \( P, Q \in E_1 \) and \( \phi(P), \phi(Q) \in E_2 \), compute the secret isogeny, \( \phi \)

- The best known attack is based on the claw finding algorithm
Isogeny-Based Cryptography underlying security

Consider two supersingular elliptic curves defined over a large prime extension field

- $E_1 / \mathbb{F}_p$ and $E_2 / \mathbb{F}_p$, where $p$ is a large prime
- There exists some isogeny $\phi : E_1 \rightarrow E_2$ with a fixed, smooth degree that is public

### Supersingular Isogeny Problem

Given $P, Q \in E_1$ and $\phi(P), \phi(Q) \in E_2$, compute the secret isogeny, $\phi$

- The best known attack is based on the **claw finding algorithm**
- For SIDH/SIKE:
Isogeny-Based Cryptography underlying security

Consider two supersingular elliptic curves defined over a large prime extension field

- $E_1/\mathbb{F}_{p^2}$ and $E_2/\mathbb{F}_{p^2}$, where $p$ is a large prime
- There exists some isogeny $\phi : E_1 \to E_2$ with a fixed, smooth degree that is public

**Supersingular Isogeny Problem**

Given $P, Q \in E_1$ and $\phi(P), \phi(Q) \in E_2$, compute the secret isogeny, $\phi$

- The best known attack is based on the claw finding algorithm
- For SIDH/SIKE:
  - Classical attack $O(p^{1/4})$
Isogeny-Based Cryptography underlying security

Consider two supersingular elliptic curves defined over a large prime extension field:
- $E_1/\mathbb{F}_p^2$ and $E_2/\mathbb{F}_p^2$, where $p$ is a large prime
- There exists some isogeny $\phi : E_1 \rightarrow E_2$ with a fixed, smooth degree that is public

**Supersingular Isogeny Problem**

Given $P, Q \in E_1$ and $\phi(P), \phi(Q) \in E_2$, compute the secret isogeny, $\phi$

- The best known attack is based on the **claw finding algorithm**
- For SIDH/SIKE:
  - Classical attack $O(p^{1/4})$
  - Quantum attack $O(p^{1/6})$
Consider a graph where each node represents *supersingular isomorphism classes*.
Visualizing the Supersingular Isogeny Problem

For $\ell = 2$, each node is connected by three unique 2-isogenies.
Consider finding an isogeny from Class A to Class B when $\ell = 2$.
Visualizing the Supersingular Isogeny Problem

For large isogeny graphs (i.e., $p$ is 512 bits or more), finding an isogeny path is HARD.

Isomorphism Class 2-isogeny

Isomorphism Class

2-isogeny
The SIDH protocol resembles standard Diffie-Hellman

\[
\begin{align*}
\text{Diffie-Hellman} & \\
\text{Alice} & \quad \text{Bob} \\
A & \quad g, p & B \\
\text{\(g^A \mod p\)} & \quad \text{\(g^B \mod p\)} \\
\text{\((g^B)^A \mod p\)} & \quad \text{\((g^A)^B \mod p\)} \\
\text{\(g^{BA} \mod p\)} & \equiv \quad \text{\(g^{AB} \mod p\)}
\end{align*}
\]
The SIDH protocol resembles standard Diffie-Hellman

**Diffie-Hellman**

<table>
<thead>
<tr>
<th>Alice</th>
<th>Bob</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A$</td>
<td>$g, p$</td>
</tr>
<tr>
<td>$g^A \mod p$</td>
<td>$g^B \mod p$</td>
</tr>
<tr>
<td>$(g^B)^A \mod p$</td>
<td>$(g^A)^B \mod p$</td>
</tr>
<tr>
<td>$g^{BA} \mod p$</td>
<td>$g^{AB} \mod p$</td>
</tr>
</tbody>
</table>

**Supersingular Isogeny Diffie-Hellman**

<table>
<thead>
<tr>
<th>Alice</th>
<th>Bob</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\phi_A$</td>
<td>$E_0/\mathbb{F}_p^2$</td>
</tr>
<tr>
<td>$\phi_A : E_0 \to E_A$</td>
<td></td>
</tr>
<tr>
<td>$E_0/\langle A \rangle$</td>
<td></td>
</tr>
<tr>
<td>$\phi'<em>A : E_B \to E</em>{BA}$</td>
<td></td>
</tr>
<tr>
<td>$E_{BA} = E_0/\langle B, A \rangle$</td>
<td></td>
</tr>
<tr>
<td>$\phi_B : E_A \to E_{AB}$</td>
<td></td>
</tr>
<tr>
<td>$E_{AB} = E_0/\langle A, B \rangle$</td>
<td></td>
</tr>
<tr>
<td>$j(E_{BA}) \mod p$</td>
<td>$j(E_{AB}) \mod p$</td>
</tr>
</tbody>
</table>
SIDH Protocol

$E_A = E_0 / \langle A \rangle$

$E_0 / \langle [m_A]P_A + [n_A]Q_A \rangle$
SIDH Protocol

\[ E_A = E_0 / \langle A \rangle \]

\[ E_B = E_0 / \langle B \rangle \]
SIDH Protocol

\[ E_A = E_0 / \langle A \rangle \]
\[ \{ \phi_A(P_B), \phi_A(Q_B) \} \]

\[ E_B = E_0 / \langle B \rangle \]
\[ \{ \phi_B(P_A), \phi_B(Q_A) \} \]
SIDH Protocol

\[ E_A = E_0 / \langle A \rangle \]

\[ \{ \phi_A(P_B), \phi_A(Q_B) \} \]

\[ E_B = E_0 / \langle B \rangle \]

\[ \{ \phi_B(P_A), \phi_B(Q_A) \} \]
SIDH Protocol

\[
E_A = E_0 / \langle A \rangle
\]

\[
E_B = E_0 / \langle B \rangle
\]

\[
\{ \phi_A(P_B), \phi_A(Q_B) \}
\]

\[
\{ \phi_B(P_A), \phi_B(Q_A) \}
\]
SIDH Computations

Secret Kernel Generation

- Inputs:
  - Supersingular elliptic curve $E(\mathbb{F}_{p^2})$, torsion basis $\{P,Q\}$, private keys $m, n$
- Compute $R = \langle [m]P + [n]Q \rangle$

Large-Degree Isogeny

- Inputs:
  - Supersingular elliptic curve $E$, secret kernel point $R$
- Compute $\phi : E \rightarrow E/\langle R \rangle$ by iteratively computing isogenies

**Figure:** Breakdown of supersingular isogeny computations
Visualizing the large-degree isogeny computation

- Large-degree isogenies can be computed by iteratively computing small-degree isogenies
- Set $E_0 = E$ and $R_0 = \ker(\phi)$
- Find kernel point $\ker(\phi_i) = \ell^{e-i-1}R_i$
- Compute $i$th isogeny $\phi_i : E_i \rightarrow E_i/\langle \ell^{e-i-1}R_i \rangle = E_{i+1}$
- Push kernel point to new curve $R_{i+1} = \phi_i(R_i)$
Creating an exposure model for SIDH

- **Exposure Model** → Assessing the security of a cryptosystem if certain pieces of information are divulged
Creating an exposure model for SIDH

- **Exposure Model** → Assessing the security of a cryptosystem if certain pieces of information are divulged
- Necessary to account for weak implementations or new attacks
Creating an exposure model for SIDH

- **Exposure Model** → Assessing the security of a cryptosystem if certain pieces of information are divulged
- Necessary to account for **weak implementations** or **new attacks**
- Looking specifically at the **large-degree isogeny**
Why might intermediate values be exposed?

- Poor implementation
- New attacks on large-degree isogeny
- Cache prime and probe
- Spectre and Meltdown
- Intermediate values not cleared
- Unexpected reset
- etc. etc.
What are the exposure classes?

- **CLASS 1**: Intermediate curve is exposed

- **CLASS 2**: Intermediate kernel point is exposed
  - Some variant of the secret kernel point is leaked
  - If corresponding curve can be found, then remaining isogenies can be computed → this is very bad

- **CLASS 3**: Intermediate basis point is exposed
  - If corresponding curve is found, then isogeny decisions are revealed
  - If the corresponding curve has been found, this situation resembles the intermediate curve scenario
What are the exposure classes?

- **CLASS 1: Intermediate curve** is exposed
  - An attacker can now split the large-degree isogeny into two separate isogenies
What are the exposure classes?

- **CLASS 1:** Intermediate curve is exposed
  - An attacker can now split the large-degree isogeny into two separate isogenies
  - Worst case scenario could cut the security assumption in half
What are the exposure classes?

- **CLASS 1:** Intermediate curve is exposed
  - An attacker can now split the large-degree isogeny into two separate isogenies
  - Worst case scenario could cut the security assumption in half
- **CLASS 2:** Intermediate kernel point is exposed
What are the exposure classes?

- **CLASS 1:** Intermediate curve is exposed
  - An attacker can now split the *large-degree isogeny* into *two separate isogenies*
  - Worst case scenario could cut the security assumption in half

- **CLASS 2:** Intermediate kernel point is exposed
  - Some variant of the *secret* kernel point is leaked
What are the exposure classes?

- **CLASS 1**: Intermediate curve is exposed
  - An attacker can now split the large-degree isogeny into two separate isogenies
  - Worst case scenario could cut the security assumption in half

- **CLASS 2**: Intermediate kernel point is exposed
  - Some variant of the secret kernel point is leaked
  - If corresponding curve can be found, then remaining isogenies can be computed → this is very bad
What are the exposure classes?

- **CLASS 1:** Intermediate curve is exposed
  - An attacker can now split the large-degree isogeny into two separate isogenies
  - Worst case scenario could cut the security assumption in half

- **CLASS 2:** Intermediate kernel point is exposed
  - Some variant of the secret kernel point is leaked
  - If corresponding curve can be found, then remaining isogenies can be computed → this is very bad

- **CLASS 3:** Intermediate basis point is exposed
What are the exposure classes?

- **CLASS 1: Intermediate curve** is exposed
  - An attacker can now split the large-degree isogeny into two separate isogenies
  - Worst case scenario could cut the security assumption in half

- **CLASS 2: Intermediate kernel point** is exposed
  - Some variant of the secret kernel point is leaked
  - If corresponding curve can be found, then remaining isogenies can be computed → this is very bad

- **CLASS 3: Intermediate basis point** is exposed
  - If corresponding curve is found, then isogeny decisions are revealed
What are the exposure classes?

- **CLASS 1: Intermediate curve** is exposed
  - An attacker can now split the **large-degree isogeny** into two separate isogenies
  - Worst case scenario could cut the security assumption in half

- **CLASS 2: Intermediate kernel point** is exposed
  - Some variant of the **secret** kernel point is leaked
  - If corresponding curve can be found, then remaining isogenies can be computed → this is very bad

- **CLASS 3: Intermediate basis point** is exposed
  - If corresponding curve is found, then isogeny decisions are revealed
  - After the corresponding curve has been found, this situation resembles the intermediate curve scenario
With an exposed kernel point (CLASS 2), an attacker can find the corresponding curve and compute the remaining isogenies that compose the secret isogeny.
An exposed kernel point is a disaster as it can be used to retrieve private keys.

General attack procedure for point after $k \ell$-isogenies and $j$ point multiplications by $\ell$

Intermediate kernel point is of the form $S = \phi_{k-1:0}([\ell^j m]P + [\ell^j n]Q)$ on curve $E_k$ (CLASS 2). Original secret kernel point is of the form $R = [m]P + [n]Q$
An exposed kernel point is a disaster as it can be used to retrieve private keys.

**General attack procedure** for point after $k$ $\ell$-isogenies and $j$ point multiplications by $\ell$

Intermediate kernel point is of the form $S = \phi_{k-1:0}([\ell^j m]P + [\ell^j n]Q)$ on curve $E_k$ (CLASS 2). Original secret kernel point is of the form $R = [m]P + [n]Q$

1. **Find isogenous curve $E_k$** to find first $k$ isogenies (difficulty $O(\ell^k)$)
An exposed kernel point is a disaster as it can be used to retrieve private keys.

**General attack procedure** for point after $k$ $\ell$-isogenies and $j$ point multiplications by $\ell$

Intermediate kernel point is of the form $S = \phi_{k-1:0}([\ell^j m]P + [\ell^j n]Q)$ on curve $E_k$ (CLASS 2). Original secret kernel point is of the form $R = [m]P + [n]Q$

1. Find isogenous curve $E_k$ to find first $k$ isogenies (difficulty $O(\ell^k)$)
2. Push torsion basis through $k$ isogenies
   - $\{\phi_{k-1:0}(P), \phi_{k-1:0}(Q)\}$
An exposed kernel point is a disaster as it can be used to retrieve private keys

**General attack procedure** for point after \( k \ell \)-isogenies and \( j \) point multiplications by \( \ell \)

Intermediate kernel point is of the form \( S = \phi_{k-1:0}([\ell^j m]P + [\ell^j n]Q) \) on curve \( E_k \) (CLASS 2). Original secret kernel point is of the form \( R = [m]P + [n]Q \)

1. Find isogenous curve \( E_k \) to find first \( k \) isogenies (difficulty \( O(\ell^k) \))
2. Push torsion basis through \( k \) isogenies
   - \( \{ \phi_{k-1:0}(P), \phi_{k-1:0}(Q) \} \)
3. Perform generalized elliptic curve discrete log (simple for SIDH curves) \( \rightarrow \) result is \( k - j \) bits of private key
   - \( \phi_{k-1:0}([\ell^j m]P + [\ell^j n]Q) = \phi_{k-1:0}([m']P) + \phi_{k-1:0}([n']Q) \)
An exposed kernel point is a disaster as it can be used to retrieve private keys.

General attack procedure for point after $k\ell$-isogenies and $j$ point multiplications by $\ell$

Intermediate kernel point is of the form $S = \phi_{k-1:0}([\ell^j m]P + [\ell^j n]Q)$ on curve $E_k$ (CLASS 2). Original secret kernel point is of the form $R = [m]P + [n]Q$

1. Find isogenous curve $E_k$ to find first $k$ isogenies (difficulty $O(\ell^k)$)
2. Push torsion basis through $k$ isogenies
   - $\{\phi_{k-1:0}(P), \phi_{k-1:0}(Q)\}$
3. Perform generalized elliptic curve discrete log (simple for SIDH curves) → result is $k - j$ bits of private key
   - $\phi_{k-1:0}([\ell^j m]P + [\ell^j n]Q) = \phi_{k-1:0}([m']P) + \phi_{k-1:0}([n']Q)$
4. Perform exhaustive search on the point multiples $j$ for the rest of the key (difficulty $O(\ell^j)$)
   - Remaining secret key bits is some secret multiple of the point order:
     $$m' \equiv x \times m \mod \ell^j$$
A random pre-isogeny isomorphism protects against all but an exposed curve

- Vélu’s formulas for isogenies are deterministic for a given kernel point and curve
A random pre-isogeny isomorphism protects against all but an exposed curve

- Vélu’s formulas for isogenies are deterministic for a given kernel point and curve
- An isomorphism moves from one curve to another within an isomorphism class
A random pre-isogeny isomorphism protects against all but an exposed curve

- Vélu’s formulas for isogenies are deterministic for a given kernel point and curve
- An isomorphism moves from one curve to another within an isomorphism class
- A random isomorphism will scale the curve
A random pre-isogeny isomorphism protects against all but an exposed curve

- Vélu’s formulas for isogenies are deterministic for a given kernel point and curve
- An isomorphism moves from one curve to another within an isomorphism class
- A random isomorphism will scale the curve
  - This scaling changes the output curves of isogenies and obfuscates any points that are exposed
The pre-isogeny isomorphism obfuscates any exposed points throughout the isogeny operation.
The pre-isogeny isomorphism obfuscates any exposed points throughout the isogeny operation.

- **Protects** against CLASS 2 and CLASS 3 exposures.
A pre-isogeny isomorphism is a computationally cheap countermeasure

- For short Weierstrass curves, this requires a random number and several field operations
A pre-isogeny isomorphism is a computationally cheap countermeasure

- For short Weierstrass curves, this requires a random number and several field operations
- Major cost is generating random numbers

**Table: Cost of Pre-isogeny Isomorphism**

<table>
<thead>
<tr>
<th>Protocol</th>
<th>r</th>
<th>δ</th>
<th>l</th>
<th>M</th>
<th>S</th>
</tr>
</thead>
<tbody>
<tr>
<td>SIDH Round 1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>9</td>
<td>3</td>
</tr>
<tr>
<td>SIDH Round 2</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>SIDH Indirect Key Validation</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>15</td>
<td>5</td>
</tr>
</tbody>
</table>

- Let r be the cost to generate a random number, l be a finite-field inversion, M be a finite-field multiplication, S be a finite-field squaring, and δ be a finite-field comparison
Conclusion

- **SIDH/SIKE** are up and coming candidates for PQC standardization
Conclusion

- **SIDH/SIKE** are up and coming candidates for **PQC standardization**
  - These primitives feature the **smallest known public key sizes** for quantum-resistant PKC
Conclusion

- SIDH/SIKE are up and coming candidates for PQC standardization
  - These primitives feature the smallest known public key sizes for quantum-resistant PKC
- We created an exposure model for the large-degree isogeny computation
Conclusion

- **SIDH/SIKE** are up and coming candidates for PQC standardization
  - These primitives feature the smallest known public key sizes for quantum-resistant PKC
- We created an exposure model for the large-degree isogeny computation
- We showed that an exposed intermediate kernel point (CLASS 2) can reveal a party’s private key
Conclusion

- SIDH/SIKE are up and coming candidates for PQC standardization
  - These primitives feature the **smallest known public key sizes** for quantum-resistant PKC
- We created an **exposure model** for the **large-degree isogeny** computation
- We showed that an **exposed intermediate kernel point** (CLASS 2) can reveal a party’s private key
- We proposed a **pre-isogeny isomorphism** as a cheap countermeasure to protect against exposing intermediate points in the future


Conclusion

- **SIDH/SIKE** are up and coming candidates for **PQC standardization**
  - These primitives feature the **smallest known public key sizes** for quantum-resistant **PKC**
- We created an **exposure model** for the **large-degree isogeny computation**
- We showed that an **exposed intermediate kernel point** (CLASS 2) can reveal a party’s private key
- We proposed a **pre-isogeny isomorphism** as a cheap countermeasure to protect against exposing intermediate points in the future
- Thank you very much for your attention. Questions?