Towards Isogeny-Based Password-Authenticated Key Establishment

Oleg Taraskin[1], Vladimir Soukharev, David Jao, and Jason T. LeGrow

[1] Waves Platform. Moscow, Russian Federation. tog.postquant@gmail.com
[2] InfoSec Global. Toronto, Ontario, Canada.
vladimir.soukharev@infosecglobal.com
[3] Department of Combinatorics and Optimization, University of Waterloo. Waterloo, Ontario, Canada. {djao,jlegrow}@uwaterloo.ca

Abstract. Password authenticated key establishment (PAKE) is a cryptographic primitive that allows two parties who share a low-entropy secret (a password) to securely establish cryptographic keys in the absence of public key infrastructure. We propose the first quantum-resistant password-authenticated key exchange scheme based on supersingular elliptic curve isogenies. The scheme is built upon supersingular isogeny Diffie-Hellman [15], and uses the password to generate permutations which obscure the auxiliary points. We include elements of a security proof, and discuss roadblocks to obtaining a proof in the BPR model [1]. We also include some performance results.

1 Introduction

Many current cryptographic schemes are based on mathematical problems that are considered difficult for classical computers, but can easily be solved using quantum algorithms. To prepare for the emergence of quantum computers, we aim to design cryptographic primitives which will resist quantum attacks. One family of such primitives, proposed by Jao and De Feo [15]—commonly referred to as SIDH—uses isogenies between supersingular elliptic curves to construct quantum-resistant cryptographic protocols for public key cryptography. Subsequent work by Costache et al. [8] has shown that the security of SIDH reduces to the Supersingular Isogeny Graph problem originally proposed by Charles et al. [7].

Password-Authenticated Key Establishment (PAKE) is a primitive in which parties securely establish a common cryptographic key over an insecure channel using a password (modelled as a low-entropy secret). The first PAKE protocol was designed by Bellovin and Merritt in 1992 [3]. Today, many protocols of this type exist—most are based on the discrete logarithm problem in subgroups of $\mathbb{Z}_p^*$ or elliptic curve groups, and are not quantum-safe. Until this work, the only PAKEs built on quantum-safe foundations are lattice-based [12, 27]. We propose the first PAKE based on isogenies between supersingular elliptic curves. It is derived from SIDH; notably, the additional operations required are not prohobitively expensive (elliptic curve arithmetic) and do not require additional rounds of communication, and the messages are the same size as in SIDH.

In particular, in SIDH, parties exchange elliptic curves which are images of a fixed public curve under ephemeral secret isogenies; then, each party computes the image of their peer's curve under a related isogeny. Parties also exchange certain points on their ephemeral curves to aid in the computation. In our PAKE protocol, these so-called "auxiliary points" are obfuscated by transforming them according to a certain group action, described in Section 2. The passive security of this protocol comes from the difficulty of finding the correct points (preventing offline dictionary attacks) or performing the computation without them, while the active security from the fact that the adversary needs to "commit," in a sense, to a group element in order to actively attack the protocol (preventing all but the most basic online dictionary attacks).

2 Background

2.1 Password-Authenticated Key Establishment

The PAKE security model of Bellare, Pointcheval, and Rogaway (BPR) [1] is very similar to the Canetti-Krawczyk (CK) security model [5] for authenticated key establishment. The complete model specification appears in Appendix A; here we only point out the major differences between the two. While the CK security model includes public-key infrastructure (*i.e.,* each party registers a keypair, and the public keys are published), in the BPR model there are two kinds of parties—clients and servers—and clients choose passwords according to some distribution on a dictionary and share them with the servers. Of course, in the real world, passwords are *low-entropy* secrets; this is typically modelled by having passwords be chosen by parties uniformly from a set called the *password dictionary* which is small enough that it can be enumerated efficiently. In particular, attacks which involve "checking all passwords" are feasible. The examples in the following section illustrate this concept.

Two Insecure PAKEs To illustrate the consequences of the low-entropy nature of passwords, we present two protocols which admit attacks that exploit it.

1. **Diffie-Hellman + MAC** Let MAC be a message authentication code protocol. Augment the Diffie-Hellman protocol by having each party send a MAC tag on each of their messages, using the password as the key; have the parties abort if any tag they receive is not consistent with the password and incoming message. Though this is reminiscent of the SIG-DH protocol of [5] (which is secure in the CK model), the low entropy of the password distribution means that it is susceptible to a straightforward *offline dictionary attack*: upon seeing a single message and tag (m, t), an adversary can simply check whether $t = \mathrm{MAC}_\pi(m)$ for each password π in the dictionary.

With overwhelming probability equality will hold for exactly one password π^*, which is the party's true password. Once the password is recovered, the adversary can successfully mount a man-in-the-middle attack.

2. **Randomized-base Diffie-Hellman** Consider a modified version of Diffie-Hellman in a group $G = \langle g \rangle$ of order N as follows. Let H be a hash function which maps the password dictionary into $(\mathbb{Z}/N\mathbb{Z})^*$. Instead of using the fixed base g, parties use base $g^{H(\pi)}$ when constructing their messages, so that if the ephemeral secret is a, the message is $m = g^{aH(\pi)}$. The shared secret is constructed as in Diffie-Hellman: if the messages are $m = g^{aH(\pi)}$ and $m' = g^{bH(\pi)}$, the shared secret is $(m')^a = g^{abH(\pi)} = m^b$. An adversary can attack the protocol by intercepting a message $m = g^{aH(\pi)}$ from Alice and inserting their own message $m'' = g^c$. Alice will compute a shared secret $s = g^{ac}$, while the adversary can compute a "potential" shared secret $s'(\pi') = m^{cH(\pi')^{-1}}$ for each password π'. By issuing a password reveal query on Alice's session, the adversary can determine Alice's true password from s and the list of $s'(\pi')$.

2.2 Isogenies

We provide a brief review of the background information. For further details on the mathematical foundations of isogenies, we refer the reader to [15, 25]. Given two elliptic curves E_1 and E_2 over some finite field $\mathbb{F}_q$ of size q, an *isogeny* $\phi : E_1 \to E_2$ is an algebraic morphism which is a group homomorphism. The degree of ϕ, denoted $\deg(\phi)$, is its degree as an algebraic morphism. Two elliptic curves are *isogenous* if there exists an isogeny between them. Given an isogeny $\phi\colon E_1 \to E_2$ of degree n, there is another isogeny $\hat{\phi}\colon E_2 \to E_1$ (the *dual* isogeny) of degree n satisfying $\phi \circ \hat{\phi} = \hat{\phi} \circ \phi = [n]$ (the multiplication-by-n map). Thus "being isogenous" is an equivalence relation. For any $n \in \mathbb{N}$, define the subgroup $E[n] = \{P \in E(\bar{\mathbb{F}}_q) : nP = \infty\}$. The group $E[n]$ is isomorphic to $(\mathbb{Z}/n\mathbb{Z})^2$ whenever $\gcd(n, q) = 1$ [25]. We define the *endomorphism ring* $\mathrm{End}(E)$ to be the set of all isogenies from E to itself, defined over the algebraic closure $\bar{\mathbb{F}}_q$ of $\mathbb{F}_q$. The endomorphism ring is a ring under the operations of point-wise addition and functional composition. If $\dim_{\mathbb{Z}}(\mathrm{End}(E)) = 2$, then we say that E is *ordinary*; otherwise $\dim_{\mathbb{Z}}(\mathrm{End}(E)) = 4$ and we say that E is *supersingular*. Two isogenous curves are either both ordinary or both supersingular. All elliptic curves used in this work are supersingular. The isogeny $\phi\colon E_1 \to E_2$ is *separable* if the extension $\mathbb{F}_q(E_1)/\phi^*(\mathbb{F}_q(E_2))$ is separable; we will only consider separable isogenies. An important property of a separable isogeny ϕ is that $|\ker \phi| = \deg \phi$ [25, III.4.10(c)]. The kernel uniquely defines the isogeny up to isomorphism. Methods for computing and evaluating isogenies are given in [4, 15, 16, 26]. We use isogenies whose kernels are cyclic; in this setting, knowledge of any single generator of the kernel allows for efficient evaluation of the isogeny (up to isomorphism); conversely, the ability to evaluate the isogeny via a black box allows for efficient determination of the kernel. Thus, in our application, the following are equivalent: knowledge of the isogeny, its kernel, or any generator of its kernel.

2.3 Isogeny-Based Cryptography

The first cryptographic construction to use supersingular elliptic curve isogenies is the hash function construction of Charles et al. [7], based on the isogeny graph path-finding problem. Public key cryptography based on isogenies was first introduced by Couveignes [10] and Rostovtsev & Stolbunov [23] using ordinary elliptic curves. Jao and De Feo [15] proposed the first public-key cryptosystem based on supersingular elliptic curve isogenies, along with a new assumption which was subsequently shown [8] to reduce to the path-finding assumption originally proposed in [7]. We review here the operation of the Jao and De Feo key exchange protocol SIDH, which is the foundation for our PAKE.

Fix a prime p of the form $\ell_A^{e_A}\ell_B^{e_B}f \pm 1$ where ℓ_A, ℓ_B are small primes, $e_A, e_B \in \mathbb{N}$ such that $\ell_A^{e_A} \approx \ell_B^{e_B}$, and $f \in \mathbb{N}$ is a small cofactor. Fix a supersingular curve E defined over $GF(p^2)$, and bases $\{P_A, Q_A\}$ and $\{P_B, Q_B\}$ of $E[\ell_A^{e_A}]$ and $E[\ell_B^{e_B}]$ respectively. Alice chooses $m_A, n_A \in_R \mathbb{Z}/\ell_A^{e_A}\mathbb{Z}$, not both divisible by ℓ_A, and computes an isogeny $\phi_A \colon E \to E_A$ with kernel $\langle m_A P_A + n_A Q_A\rangle$. Alice computes the auxiliary points $\{\phi_A(P_B), \phi_A(Q_B)\} \subset E_A$, and sends these points to Bob along with E_A. Bob proceeds analogously. Upon receipt of E_B and $\phi_B(P_A), \phi_B(Q_A) \in E_B$ from Bob, Alice computes an isogeny $\phi'_A \colon E_B \to E_{AB}$ with kernel $\langle m_A\phi_B(P_A) + n_A\phi_B(Q_A)\rangle$; Bob proceeds *mutatis mutandis*. Alice and Bob can then form a shared secret key using use the common j-invariant of

$$E_{AB} = \phi'_B(\phi_A(E)) = \phi'_A(\phi_B(E)) = E/\langle m_A P_A + n_A Q_A, m_B P_B + n_B Q_B\rangle.$$

With almost no loss in generality, private ephemeral keys with $m = 1$ can be used [9]. In this work we follow that convention.

2.4 The Möbius Action and Auxiliary Point Obfuscation

For a prime ℓ and an integer e, we define

$$\begin{aligned} \mathrm{SL}_2(\ell, e) &= \left\{\varPsi \in (\mathbb{Z}/\ell^e\mathbb{Z})^{2\times 2} \,:\, \det A \equiv 1 \pmod{\ell^e}\right\} \\ \varUpsilon_2(\ell, e) &= \{\varPsi \in \mathrm{SL}_2(\ell, e) \,:\, A \text{ is upper triangular modulo } \ell\} \end{aligned}$$

to be the special linear and special reduced upper triangular groups modulo ℓ^e, respectively. If $p = \ell_A^{e_A}\ell_B^{e_B}f \pm 1$ is a prime and E is a supersingular elliptic curve defined over $GF(p^2)$, $\varUpsilon_2(\ell_A, e_A)$ acts on $E[\ell_A^{e_A}]^2$ in a way analogous with ordinary matrix-vector multiplication: if $\varPsi = \left[\begin{smallmatrix}\alpha & \beta \\ \gamma & \delta\end{smallmatrix}\right]$ then $\varPsi\left[\begin{smallmatrix}X \\ Y\end{smallmatrix}\right] = \left[\begin{smallmatrix}\alpha X + \beta Y \\ \gamma X + \delta Y\end{smallmatrix}\right]$. We use this action to mask auxiliary points, since for any $X, Y \in E[\ell_A^{e_A}]$ and any $\varPsi, \varPsi' \in \varUpsilon_2(\ell_A, e_A)$, for $\left[\begin{smallmatrix}X' \\ Y'\end{smallmatrix}\right] = \varPsi^{-1}\varPsi'\left[\begin{smallmatrix}X \\ Y\end{smallmatrix}\right]$ we have $e(X, Y) = e(X', Y')$, where e is the Tate pairing [24, Exercise 10.24]; this prevents a kind of offline dictionary attack, which we elaborate on in Section 4.1.

In the context of isogeny computations, this (left) action of $\varUpsilon_2(\ell_A, e_A)$ induces a new (right) action (the *Möbius* action) on $\mathbb{Z}/\ell_A^{e_A}\mathbb{Z}$ in the following way: if $\left[\begin{smallmatrix}X' \\ Y'\end{smallmatrix}\right] = \varPsi\left[\begin{smallmatrix}X \\ Y\end{smallmatrix}\right]$ where $\varPsi = \left[\begin{smallmatrix}\alpha & \beta \\ \gamma & \delta\end{smallmatrix}\right]$, then for any $m \in \mathbb{Z}/\ell_A^{e_A}\mathbb{Z}$

$$E/\langle X' + mY'\rangle = E/\langle(\alpha X + \beta Y) + m(\gamma X + \delta Y)\rangle = E/\langle X + \tfrac{\delta m + \beta}{\gamma m + \alpha}Y\rangle;$$

thus we define the right action $m^{\Psi} := \frac{\delta m+\beta}{\gamma m+\alpha}$. The two actions are related as above: mapping the auxiliary basis by the action of Ψ on $E[\ell_A^{e_A}]^2$ is equivalent to mapping the m which defines the isogeny's kernel by the action of Ψ on $\mathbb{Z}/\ell_A^{e_A}\mathbb{Z}$. This action relates to a sort of "related-key attack" on SIDH: in an unauthenticated setting, an active adversary can force an honest party to compute a shared key corresponding to an ephemeral key chosen by the adversary and the image of the honest party's ephemeral key under the action of a known matrix Ψ.

Note that given E_A with $E_A = E_K/\langle P_A + m_A Q_A\rangle$ and $\Psi \in \Upsilon_2(\ell_A, e_A)$ it is easy to find $(E_B, X_B, Y_B), E_{AB,\Psi}$ satisfying $E_{AB,\Psi} = E_B/\langle X_B + m_A^{\Psi} Y_B\rangle$; simply set $E_B = E_K/\ker\phi$ for some $\ell_B^{e_B}$-isogeny ϕ, and set $[X_B, Y_B]^T = \Psi^{-1}[\phi(P_B), \phi(Q_B)]^T$.

2.5 Computational Assumptions

For brevity, the standard assumptions of SIDH appear in Appendix B.

As noted in Section 2.3, to compute the shared secret in SIDH the parties must have the images of the public torsion bases under the secret isogenies. While previous works ignored these "auxiliary points" in favour of standard authentication methods, such as signature schemes [18] or generic transforms [14] to add authentication, here we focus on how we can disrupt man-in-the-middle attacks by obfuscating them. In this section we define a new computational assumption related to computing these auxiliary points. We use the notation of Section 2.3 for our global parameters, and define the security parameter $\lambda = \lceil \log_2 p \rceil$. When P is a computational problem with some global parameters (such as the prime p and global curves and torsion points used in SIDH), we denote by $\texttt{Ins}_P(\gamma)$ the set of all instances of problem P with global parameters γ. When $\mathcal{A}$ is an algorithm for a problem P and ϑ is an instance of P, $\mathcal{A}(\vartheta)$ denotes the (possibly randomized) output of $\mathcal{A}$ on input ϑ. $\mathrm{Unif}(X)$ denotes the uniform distribution on a set X.

The asymmetry of the isogeny computations requires us to consider two variants of each of the computational problem: one for $\ell_A^{e_A}$-isogenies, and the other for $\ell_B^{e_B}$-isogenies. For brevity we present the "Type A" problem, advantage, and assumption here and omit the "Type B" variant, which is defined analogously.

Definition 1 (Auxillary Point Computation–A (SI-APC$_A$)). *Let $\phi_A\colon E \to E_A$ be an $\ell_A^{e_A}$-isogeny with kernel $\langle P_A + n_A Q_A\rangle$ for $n_A \leftarrow \mathrm{Unif}(\mathbb{Z}/\ell_A^{e_A}\mathbb{Z})$. The supersingular isogeny auxiliary point computation problem (type A) is to compute $\phi_A(P_B)$ and $\phi_A(Q_B)$ given $(E, P_A, Q_A, P_B, Q_B, E_A)$.*

With the notation above, we define $\text{SI-APC}_A((E, P_A, Q_A, P_B, Q_B, E_A)) = (\phi_A(P_B), \phi_A(Q_B))$. For algorithms $\mathcal{A}$ which solve SI-APC$_A$, we define the advantages

$$\mathrm{Adv}_p^{\text{SI-APC}_A}(\mathcal{A}) = \mathbb{P}[\vartheta \leftarrow \texttt{Ins}_{\text{SI-APC}_A}(p) : \text{SI-APC}_A(\vartheta) \in \mathcal{A}(\vartheta)], \text{ and}$$
$$\mathrm{Adv}_p^{\text{SI-APC}_A}(t) = \max\{\mathrm{Adv}_p^{\text{SI-APC}_A}(\mathcal{A}) : \mathcal{A} \text{ runs in time } t\ \forall \vartheta \in \texttt{Ins}_{\text{SI-APC}_A}(p)\}.$$

The SI-APC assumption is that for t and n polynomial in λ, $\mathrm{Adv}_p^{\text{SI-APC}_A}(t, n)$ and $\mathrm{Adv}_p^{\text{SI-APC}_B}(t, n)$ are negligible in λ.

Of course, the MOV attack [20] attack using the Pohlig-Hellman algorithm [22] can be used to solve extended discrete logarithms in supersingular elliptic curves, and so the SI-APC problem is equivalent to asking the adversary to find the image of *any* $\ell_A^{e_B}$- (or $\ell_B^{e_B}$-) torsion point—rather than the image of a particular basis; this phrasing is arguably more natural. The SI-APC problem reduces to the supersingular isogeny problem [11, Problem 5.2] by noting that finding a generator of the kernel of an isogeny ϕ allows one to compute the isogeny on the whole domain curve, by Vélu's formulas [26], whereas SI-APC require one to compute the isogeny on particular points (or equivalently, the restriction $\phi|_{E[\ell_A^{e_A}]}$ or $\phi|_{E[\ell_B^{e_B}]}$.)

The hardness of Problem 1 has not been thoroughly studied; however, we note that if the problem could be solved efficiently, the auxiliary points needed in SIDH [11] would not have to be sent, and thus the bandwidth of SIDH could be reduced by omitting them. So far, there are no proposals for how to do away with auxiliary points in SIDH, and no known solutions to the SI-APC problems. Notably, however, it is shown in [21] that, under some heuristics, SI-APC is equivalent to the corresponding isogeny problems for some non-standard variant SIDH parameter sets (*e.g.*, when $\ell_A^{e_A} \gg \ell_B^{e_B}$, or *vice versa*).

3 Our Protocol

The protocol builds on the SIDH scheme of Jao and De Feo [15]. Suppose a client $A \in \mathcal{C}$ and a server $B \in \mathcal{S}$ who share a common password π_A wish to establish a shared secret key. The setup is as follows: Fix a prime p of the form $\ell_A^{e_A}\ell_B^{e_B} \cdot f \pm 1$ where ℓ_A and ℓ_B are small primes, e_A and e_B are positive integers, and f is a (small) cofactor. Fix a supersingular curve E_K defined over $GF(p^2)$, and bases $\{P_A, Q_A\}$ and $\{P_B, Q_B\}$. Let H_A and H_B be random oracles with codomain$(H_A) = \Upsilon_2(\ell_B, e_B)$ and codomain$(H_B) = \Upsilon_2(\ell_A, e_A)$, and let KDF be a key derivation function (modelled as a random oracle). Then:

1. A chooses an ephemeral key and a message to B. In particular, A:
   (a) Chooses $n_A \in \mathbb{Z}_{\ell_A^{e_A}}$ uniformly and sets $G_A = P_A + n_A Q_A$;
   (b) Defines $E_A = E_K/\langle G_A\rangle$ and ϕ_A to be the isogeny with kernel $\langle G_A\rangle$;
   (c) Defines $\Psi_A = H_A(j(E_A)||\pi_A)$ and sets $\left[\begin{smallmatrix} X_A \\ Y_A \end{smallmatrix}\right] = \Psi_A \cdot \left[\begin{smallmatrix} \phi_A(P_B) \\ \phi_A(Q_B) \end{smallmatrix}\right]$; and,
   (d) Sends (E_A, X_A, Y_A) to B.
2. Upon receiving (E_A, X_A, Y_A) from A, B:
   (a) Checks that $e_{E_A}(X_A, Y_A) = e_{E_K}(P_B, Q_B)^{\ell_A^{e_A}}$—if not, abort;
   (b) Computes $\Psi_{AB} = H_A(j(E_A)||\pi_A)$;
   (c) Chooses $n_B \in \mathbb{Z}_{\ell_B^{e_B}}$ uniformly and sets $G_B = P_B + n_B Q_B$;
   (d) Defines $E_B = E_K/\langle G_B\rangle$ and ϕ_B to be the isogeny with kernel $\langle G_B\rangle$;
   (e) Computes $\Psi_B = H_B(j(E_B)||\pi_A)$ and sets $\left[\begin{smallmatrix} X_B \\ Y_B \end{smallmatrix}\right] = \Psi_B \cdot \left[\begin{smallmatrix} \phi_B(P_A) \\ \phi_B(Q_A) \end{smallmatrix}\right]$;
   (f) Sends (E_B, X_B, Y_B) to A; and,
   (g) Constructs the key K_B, which is given by

$$\mathrm{KDF}((E_A, X_A, Y_A)||(E_B, X_B, Y_B)||j(E_A/\langle X_A + n_B^{\Psi_{AB}^{-1}} Y_A\rangle)||\Psi_{AB}||\Psi_B)$$

3. Upon receiving (E_B, X_B, Y_B) from B, A:
   (a) Checks that $e_{E_B}(X_B, Y_B) = e_{E_K}(P_A, Q_A)^{\ell_B^{e_B}}$—if not, abort;
   (b) Computes $\Psi_{BA} = H_B(j(E_B)||\pi_A)$; and,
   (c) Constructs the key K_A, given by

$$\mathrm{KDF}((E_A, X_A, Y_A)||(E_B, X_B, Y_B)||j(E_B/\langle X_B + n_B^{\Psi_{BA}^{-1}} Y_B\rangle)||\Psi_A||\Psi_{BA})$$

From the definitions of K_A and K_B, the correctness of the protocol follows if $j(E_A/\langle X_A + n_B^{\Psi_{AB}^{-1}} Y_A\rangle) = j(E_B/\langle X_B + n_B^{\Psi_{BA}^{-1}} Y_B\rangle)$. We have

$$\begin{aligned} E_A/\langle X_A + n_B^{\Psi_{AB}^{-1}} Y_A\rangle &= (E_K/\ker\phi_A)/\langle \phi_A(P_B) + n_B\phi_A(Q_B)\rangle \\ &\cong (E_K/\ker\phi_A)/\phi_A(\ker\phi_B) \\ &\cong E_K/\langle \ker\phi_A, \ker\phi_B\rangle \\ &\cong (E_K/\ker\phi_B)/\phi_B(\ker\phi_A) \\ &\cong (E_K/\ker\phi_B)/\langle \phi_B(P_A) + n_A\phi_B(Q_A)\rangle \\ &= E_B/\langle X_B + n_A^{\Psi_{BA}^{-1}} Y_B\rangle \end{aligned}$$

so that $j(E_A/\langle X_A + n_B^{\Psi_{AB}^{-1}} Y_A\rangle) = j(E_B/\langle X_B + n_B^{\Psi_{BA}^{-1}} Y_B\rangle)$ (since the j-invariant is isomorphism-invariant [24, Proposition 1.4.(b)]), and the protocol is correct.

4 Progress Toward—and Roadblocks to—a Security Theorem

Ideally, we would like a security theorem of the following form: for all adversaries $\mathcal{A}$ which run in time t and use n_S, n_E, n_O queries to `Send`, `Execute`, and the random oracles respectively, we have

$$\begin{aligned} \mathrm{Adv}_{\Pi}^{\Gamma}(\mathcal{A}) \leq \frac{\alpha n_S}{L} &+ \mathrm{poly}(n_S, n_E, n_O, t)\cdot \mathrm{negl}(\lambda) \\ &+ \sum_j \alpha_j \mathrm{Adv}^{P_j}(\mathrm{poly}(t, n_S, n_E, n_O)) \end{aligned} \tag{1}$$

where $\alpha \geq 1$, the terms $\mathrm{Adv}^{P_j}(\mathrm{poly}(t, n_S, n_E, n_O))$ encode the success probability in solving the underlying computational problems. Intuitively, this says "the protocol is secure, up to terms related to an adversary's ability to solve the underlying computational problems plus a negligible probability, and up to online dictionary attacks which allow α password guesses per online session on average."

In this section we present partial results toward a security theorem of this form for the protocol of Section 3, and discuss the roadblocks that have prevented us from establishing a complete security proof.

4.1 Successes

A birthday-type bound. The adversary can break the security of the protocol if, by chance, two sessions use the same ephemeral secret key. A straightforward birthday bound argument demonstrates that this occurs with probability at most

$$\frac{2(n_S + n_E)(n_S + n_E + n_O + n_K)}{\min\{\ell_A^{e_A}, \ell_B^{e_B}\}} = \text{poly}(n_S, n_E, n_O, t) \cdot \text{negl}(\lambda).$$

Preventing offline dictionary attacks from SIDH public key validation. The auxiliary points $(\phi_A(P_B), \phi_A(Q_B))$ (respectively, $(\phi_B(P_A), \phi_B(Q_A))$) used in SIDH are determined by the ephemeral secret key, and hence are information-theoretically determined by the public ephemeral curve E_A (respectively, E_B). In [13], the authors demonstrate an attack on static-ephemeral SIDH which uses maliciously-generated false auxiliary points to determine one party's static public key; in response, auxiliary point verification procedures have been developed. The most robust known verification measure—present in [9]—ensures that the Tate pairing of the auxiliary points is correct; that is, one checks whether

$$e_{E_A}(X_A, Y_A) = e_{E_K}(P_B, Q_B)^{\ell_A^{e_A}}.$$

This has the potential to yield an offline dictionary attack against protocols which obfuscate auxiliary points, as follows: upon receiving (E_A, X_A, Y_A) from a client whose password is π, the adversary constructs $\Psi(\pi') = H_A(E_A||\pi')$ for each password π' in the dictionary, and then constructs each pair $[R_A(\pi'), S_A(\pi')]^T = \Psi(\pi')^{-1}[X_A, Y_A]$ of points. Then, the adversary checks if

$$e_{E_A}(R_A(\pi'), S_A(\pi')) = e_{E_K}(P_B, Q_B)^{\ell_A^{e_A}} \tag{2}$$

for each password π'. For any π' for which Equation 2 does *not* hold, the adversary is certain that the client's password is not π', since the true auxiliary points $R_A(\pi)$ and $S_A(\pi)$ will satisfy it. This could allow the adversary to cut down the set of "possible passwords," without needing to launch an active attack.

Our choice of auxiliary point obfuscation method is not susceptible to this attack since for all $\Psi' \in \Upsilon_2(\ell_A, e_A)$ we have

$$e_{E_K}\left(\Psi'^{-1}[X_A, Y_A]^T\right) = e_{E_K}(X_A, Y_A)^{\det \Psi'^{-1}} = e_{E_K}(X_A, Y_A);$$

in particular, the pairing value is the same for each pair of points $R'_A(\pi), S'_A(\pi)$. The same argument applies to Bob's messages, and so this particular offline dictionary attack is thwarted.

Enforcing knowledge of the auxiliary points. In SIDH, the shared secret is $j(E/\langle G_A, G_B\rangle)$, where G_A and G_B are the ephemeral secret kernel generators, and the (passive) security of the protocol is predicated only on an adversary's inability to compute this quantity from the messages. Notably, the shared secret is determined (information-theoretically) by E_A and E_B alone; the auxiliary points are required only to make the computation efficient. In contrast, in order for our protocol to be secure against offline dictionary attacks, it is necessary to assume that it is difficult to recover not only the SIDH shared secret from E_A and E_B, but also the correct auxiliary points. The most natural idea is to make the SI-APC assumption as described in Section 2.5; unfortunately, if the session key depends only on the SIDH shared secret, we cannot extract a solution to an SI-APC instance from an adversary who wins the security game, and thus cannot relate the security of the protocol to the SI-APC assumption directly.

The solution: we must include auxiliary point information in the keying information. By including Ψ_A and Ψ_B as arguments to KDF, we can extract auxiliary points from the random oracle inputs of an adversary that wins the security game, and solve an instance of SI-APC. To solve an instance of SI-APC$_A$ we can insert the instance into a client's message, choosing the auxiliary points at random. With probability $[\mathrm{SL}_2(\ell_B, e_B) : \Upsilon_2(\ell_B, e_B)]^{-1} = (\ell_B + 1)^{-1}$ the randomly-chosen points are "valid" (in the sense that there exists an element of $\Upsilon_2(\ell_B, e_B)$ that takes the true auxiliary points to the randomly-chosen points). Any adversary who wins the game with non-negligible advantage must make a KDF query with the correct Ψ_A and Ψ_B values, since KDF is a random oracle; thus any such adversary must obtain the correct Ψ_A and Ψ_B either by

1. Querying H_A and H_B on the correct values, or;
2. Extracting them from the publicly-available, password-related information.

In the second case we can extract solutions to SI-APC$_A$ (respectively SI-APC$_B$), by using the Ψ_A (respectively, Ψ_B) value from the adversary's correct KDF query.

4.2 Roadblocks

Password information in messages. In classical PAKE protocols, the message distribution is typically independent of the users' passwords; in contrast, in our protocol the password is information-theoretically determined by a given message, but extracting the information from the messages is assumed to be difficult. Nevertheless, the possibility of obtaining partial password information from the messages alone cannot easily be ruled out, and so it is very difficult to quantify the number of passwords that can be eliminated per actively-attacked session (the α in equation (1)). Notably, some other post-quantum PAKEs, such as RLWE-PAK and RLWE-PPK [12] also have message distributions which are not independent of the password.[4]

[4] In both schemes, messages are of the form $m = H(\pi) + a \cdot s + 2e$ where $(a, a \cdot s + 2e)$ is a RLWE sample [19, Definition 3.1]. For passwords $\pi' \neq \pi$, $m - H(\pi')$ is distributed uniformly, while $m - H(\pi)$ is distributed according to the RLWE distribution. Though it is generally assumed that these distributions are computationally indistinguishable [19, Definition 3.2], they are not *equal*; *i.e.,* the messages are not statistically independent of the password.

Password guesses. A typical PAKE security argument defines the notion of "password guess," and then argues that at most α password guesses can be made per session, under some computational assumption. A password guess on password π is intuitively "a KDF query in which the messages are those messages sent in a pair of matching sessions, the shared secret is the one that would be computed in the pair of matching sessions if the password were π, and the other inputs to KDF are consistent with the password and (if appropriate) have been the output of random oracle queries with password π."

A natural-seeming notion for password guess on a client is that, for values $(E, X, Y), (E', X', Y'), j(E''), \Psi, \Psi'$ the adversary $\mathcal{A}$ queries

- $\mathrm{KDF}((E, X, Y)||(E', X', Y')||j(E'')||\Psi||\Psi')$;
- $\texttt{Send}(A, i, (\texttt{Initiate}, B))$ with output (E, X, Y);
- $\texttt{Send}(A, i, (E', X', Y'))$;
- $H_A(j(E)||\pi) = \Psi$ and $H_B(j(E')||\pi) = \Psi'$

where $E'' \cong E'/\langle X' + n_A^{\Psi'^{-1}} Y'\rangle$, where n_A is the secret key which yields E. This is somewhat consistent with our intuition of what a password guess should be: $\mathcal{A}$ executes the protocol with A (pretending to be B) and computes the key that A would compute if her password were π. While this does model one form of password guess, the asymmetry of the protocol means that there is another form of password guess which is consistent with the messages of the protocol: by sending (E', X', Y') where $E' = \langle P_B + n_B Q_B\rangle$, X', Y' are the images of the true auxiliary points under the action of $\Psi'' = H_B(j(E')||\pi'')$ and then computing $\mathrm{sk}'' = \mathrm{KDF}((E, X, Y)||(E', X', Y')||E/\langle X + n_B^{\Psi^{-1}} Y\rangle||\Psi||\Psi'')$ (where $\Psi = H_A(j(E)||\pi''))$, the adversary can test whether Ψ'' is A's true password by revealing A's session key sk and testing whether $\mathrm{sk}'' = \mathrm{sk}$. This type of password guess creates two difficulties for a security proof:

1. It is undetectable without knowledge of the ephemeral key n_B, and;
2. It is difficult to formulate a computational problem whose hardness can be used to bound the number of passwords that can be checked per session.

If the "natural" password guess were the only type, under the heuristic assumption that the messages reveal no password information, we can show that only one password can be guessed per session (and, furthermore, that the protocol is secure in the model of [1], under further assumptions) under the assumption that the following computational problem (and its B-type variant) is difficult:

Definition 2 (Computational Simultaneous Group Action Problem—(Type A) (C-SGA$_A$)). *Let $\phi_A : E \to E_A$ be an isogeny with kernel $\langle P_A + m_A Q_A\rangle$ for $m_A \leftarrow \mathrm{Unif}(\mathbb{Z}/\ell_A^{e_A}\mathbb{Z})$, and let $\Psi_1, \Psi_2 \leftarrow \mathrm{Unif}(\Upsilon_2(\ell_B, e_B))$. The Computational Simultaneous Group Action Problem (Type A) (C-SGA$_A$) is to find values $(E_B, X_B, Y_B), E_1, E_2$ which satisfy $E_1 = E_B/\langle X_B + m_A^{\Psi_1} Y_B\rangle$ and $E_2 = E_B/\langle X_B + m_A^{\Psi_2} Y_B\rangle$ given (E, P_B, Q_B) and $(E_A, \phi_A(P_B), \phi_A(Q_B))$.*

5 Performance

It is clear from the protocol description in Section 3 that the message sizes in our protocol are identical to those of SIDH [11] for the same parameter set; in particular, the are among the smallest post-quantum message sizes at equivalent security levels. We implemented the scheme for the two parameter sets `p434` and `p503` from [14] to quantify the additional computational cost due to auxiliary point obfuscation/unobfuscation. Table 1 contains our performance results.

Parameter Set	Scheme	Total Clock cycles ($\times 10^6$)
`p434`	SIDH	65
`p503`	SIDH	116
`p434`	Our Protocol	$65 + 77 = 142$
`p503`	Our Protocol	$116 + 112 = 228$

Table 1. Quantitative results for our protocol compared with unauthenticated SIDH. Results measured on a machine running Ubuntu 18.04 LTS with a 1.6 GHz Intel Core i5-8250U processor.

6 Conclusion

We have presented a proposal for an isogeny-based password-authenticated key establishment protocol based on supersingular isogeny Diffie-Hellman. Of particular interest is that our protocol explicitly makes use of the auxiliary points for *security*, rather than *efficiency*. We hope that the partial results presented here can serve as a stepping stone on the path to provably-secure isogeny-based PAKE.

References

1. Mihir Bellare, David Pointcheval, and Phillip Rogaway. Authenticated key exchange secure against dictionary attacks. In Bart Preneel, editor, *Advances in Cryptology — EUROCRYPT 2000*, pages 139–155, Berlin, Heidelberg, 2000. Springer Berlin Heidelberg.
2. Mihir Bellare and Phillip Rogaway. Entity authentication and key distribution. In Douglas R. Stinson, editor, *Advances in Cryptology — CRYPTO '93*, pages 232–249, Berlin, Heidelberg, 1994. Springer Berlin Heidelberg.
3. Steven M. Bellovin and Michael Merritt. Encrypted key exchange: Password-based protocols secure against dictionary attacks. In *IEEE Symposium On Research In Security And Privacy*, pages 72–84, 1992.
4. Reinier Bröker, Denis Charles, and Kristin Lauter. Evaluating large degree isogenies and applications to pairing based cryptography. In *Proceedings of the 2nd International Conference on Pairing-Based Cryptography*, pages 100–112, 2008.

5. Ran Canetti and Hugo Krawczyk. Analysis of key-exchange protocols and their use for building secure channels. In Birgit Pfitzmann, editor, *EUROCRYPT*, volume 2045 of *Lecture Notes in Computer Science*, pages 453–474. Springer, 2001.
6. Ran Canetti and Hugo Krawczyk. Analysis of key-exchange protocols and their use for building secure channels. In Birgit Pfitzmann, editor, *Advances in Cryptology—EUROCRYPT 2001*, pages 453–474, Berlin, Heidelberg, 2001. Springer.
7. Denis Xavier Charles, Kristin E. Lauter, and Eyal Z. Goren. Cryptographic hash functions from expander graphs. *Journal of Cryptology*, 22:93–113, 2009.
8. Anamaria Costache, Brooke Feigon, Kristin Lauter, Maike Massierer, and Anna Puskas. Ramanujan graphs in cryptography. Research Directions in Number Theory: Women in Numbers IV, AWM Springer Series (to appear), 2019. https://eprint.iacr.org/2018/593.
9. Craig Costello, Patrick Longa, and Michael Naehrig. Efficient algorithms for supersingular isogeny diffie-hellman. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology – CRYPTO 2016: 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part I*, pages 572–601, Berlin, Heidelberg, 2016. Springer Berlin Heidelberg.
10. Jean-Marc Couveignes. Hard homogeneous spaces, 2006. http://eprint.iacr.org/2006/291/.
11. Luca De Feo, David Jao, and Jérôme Plût. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. *J. Math. Cryptol.*, 8:209–247, 2014.
12. Jintai Ding, Saed Alsayigh, Jean Lancrenon, Saraswathy RV, and Michael Snook. Provably secure password authenticated key exchange based on rlwe for the post-quantum world. In Helena Handschuh, editor, *Topics in Cryptology – CT-RSA 2017*, pages 183–204, Cham, 2017. Springer.
13. Steven D. Galbraith, Christophe Petit, Barak Shani, and Yan Bo Ti. On the security of supersingular isogeny cryptosystems. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *Advances in Cryptology – ASIACRYPT 2016*, pages 63–91, Berlin, Heidelberg, 2016. Springer Berlin Heidelberg.
14. David Jao, Reza Azarderakhsh, Matthew Campagna, Craig Costello, Luca De Feo, Basil Hess, Amir Jalali, Brian Koziel, Brian LaMacchia, Patrick Longa, Michael Naehrig, Geovandro Pereira, Joost Renes, Vladimir Soukharev, and David Ubanik. Supersingular isogeny key encapsulation. Technical report, NIST Post-Quantum Cryptography Standardization Process, 2019.
15. David Jao and Luca De Feo. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. In Bo-Yin Yang, editor, *PQCrypto*, volume 7071 of *Lecture Notes in Computer Science*, pages 19–34. Springer, 2011.
16. David Jao and Vladimir Soukharev. A subexponential algorithm for evaluating large degree isogenies. In *Algorithmic number theory*, volume 6197 of *Lecture Notes in Comput. Sci.*, pages 219–233. Springer, Berlin, 2010.
17. Brian LaMacchia, Kristin Lauter, and Anton Mityagin. Stronger security of authenticated key exchange. In Willy Susilo, Joseph K. Liu, and Yi Mu, editors, *Provable Security: First International Conference*, pages 1–16, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg.
18. Jason LeGrow. Post-quantum security of authenticated key establishment protocols. Master's thesis, University of Waterloo, 2016.
19. Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. In Henri Gilbert, editor, *Advances in Cryptology – EUROCRYPT 2010*, pages 1–23, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

20. Alfred J. Menezes, Tatsuaki Okamoto, and Scott A. Vanstone. Reducing elliptic curve logarithms to logarithms in a finite field. *IEEE Trans. Inform. Theory*, 39(5):1639–1646, 1993.
21. Christophe Petit. Faster algorithms for isogeny problems using torsion point images. In Tsuyoshi Takagi and Thomas Peyrin, editors, *Advances in Cryptology – ASIACRYPT 2017*, pages 330–353, Cham, 2017. Springer International Publishing.
22. S. Pohlig and M. Hellman. An improved algorithm for computing logarithms over and its cryptographic significance (corresp.). *IEEE Trans. Inf. Theor.*, 24(1):106–110, September 2006.
23. Alexander Rostovtsev and Anton Stolbunov. Public-key cryptosystem based on isogenies, 2006. http://eprint.iacr.org/2006/145/.
24. Joseph H. Silverman. *The Arithmetic of Elliptic Curves.* Number 106 in Graduate Texts in Mathematics. Springer, New York, 1986.
25. Joseph H. Silverman. *The arithmetic of elliptic curves*, volume 106 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, 1992.
26. Jacques Vélu. Isogénies entre courbes elliptiques. *C. R. Acad. Sci. Paris Sér. A-B*, 273:A238–A241, 1971.
27. Jiang Zhang and Yu Yu. Two-round pake from approximate sph and instantiations from lattices. In Tsuyoshi Takagi and Thomas Peyrin, editors, *Advances in Cryptology—ASIACRYPT 2017*, pages 37–67, Cham, 2017. Springer.

Acknowledgements. This research was undertaken thanks in part to funding from the Canada First Research Excellence Fund, CryptoWorks21, NSERC, Public Works and Government Services Canada, and the Royal Bank of Canada.

A The Security Model

PAKE security is commonly proved in the model of Bellare, Pointcheval, and Rogaway [1], which we briefly review here.

Protocols. Fundamentally, a protocol Π is a probabilistic algorithm which maps strings (in this context, concatenations of passwords, randomness, and protocol-specific messages and parameters) to strings (keys).

Parties and Party Identifiers. Participants in protocols are called *parties.* Parties are either *clients* $\mathcal{C}$ or *servers* $\mathcal{S}$. In this model, protocols are initiated by clients and responded to by servers. All clients are served by all servers. Each party $P \in \mathcal{C} \sqcup \mathcal{S}$ is uniquely identified by a string a fixed length; if P is a party, we will also use P to refer to P's identifier string as needed.

Passwords. We work in the *symmetric* password model: each client $A \in \mathcal{C}$ has a *password* π_A, and each server has a collection $\{\pi_A\}_{A \in \mathcal{C}}$.

Password Initialization. In models of authenticated key establishment, parties establish public-key/private-key pairs and securely publish their public keys (*e.g.,* [2, 6, 17]) in a pre-protocol "initialization" phase. Similarly, in this model there is a pre-protocol initialization phase where users generate passwords and "install" them on servers (*i.e.*, servers get the passwords) securely.

Party Instances. Associated to each party $U \in \mathcal{C} \sqcup \mathcal{S}$ is a collection of *party instances* $\{\Omega_U^{(n)}\}_{n \in \mathbb{N}}$. When the adversary interacts with a party he may be required to specify an instance. These instances model parties establishing keys at different times and with different partners, and these different key-establishing sessions may be attacked differently (or not at all) by the adversary.

Acceptance and Termination. A party instance "accepts" when they compute a session key, and "terminates" when it will send no more messages. In our protocol, acceptance is always followed immediately by termination, but termination may occur without acceptance (*e.g.,* in the case of ill-formed incoming messages).

The Security Experiment. Informally, in the security experiment, a new entity (the *adversary*) attempts to break semantic security of the protocol after interacting with the parties and one additional "formal" party who "administrates" the game (the *challenger*). The security experiment proceeds in three stages

1. **Initialization:** The adversary chooses disjoint sets $\mathcal{C}$ and $\mathcal{S}$ of client and server identifiers. Each client $A \in \mathcal{C}$ generates a password according to a probability distribution on their password-space: $\pi_A \leftarrow \mathcal{P}_A \ \forall A \in \mathcal{C}$; then, each server $B \in \mathcal{S}$ receives all client passwords. Password-generation happens out-of-view of the adversary, though the adversary knows each $\mathcal{P}_A$.
2. **Information Gathering:** The adversary performs computations and interacts with the parties and challenger. The computational power of the adversary is not fixed in the model, but his interactions are limited to:
   (a) `Send`(U, n, msg): Message M is sent to $\Omega_U^{(n)}$. As a result, $\Omega_U^{(n)}$'s internal state is updated according to Π.
   (b) `Reveal`(U, n): The instance $\Omega_U^{(n)}$ reveals its session key $\mathrm{sk}_U^{(n)}$ (if it exists).
   (c) `Execute`(A, n, B, m): If $A \in \mathcal{C}$, $B \in \mathcal{S}$, and neither $\Omega_A^{(n)}$ nor $\Omega_B^{(m)}$ has been used, the challenger $\mathcal{C}$ instructs $\Omega_A^{(n)}$ to execute Π with $\Omega_B^{(n)}$. The transcript of this execution is then provided to $\mathcal{A}$.
   (d) `Corrupt`(U): If $U \in \mathcal{C}$, return π_U. If $U \in \mathcal{S}$, return $\{\pi_A^U\}_{A \in \mathcal{C}}$.
   (e) `Test`(U, n): The challenger $\mathcal{C}$ chooses $b \in \{0, 1\}$ uniformly at random; if $b = 0$, the challenger reveals $\mathrm{sk}_U^{(n)}$, while if $b = 1$, the challenger chooses a string uniformly at random and reveals it.

   Remark 1. Our definition of `Corrupt` in item (d) above is known as the *weak corruption model* [1, Remark 3]. In the *strong corruption model*, a `Corrupt`(U) query also reveals the state of each oracle $\Omega_U^{(n)}, n \in \mathbb{N}$; this presents another way by which a secret session key can become known to the adversary, and the definition of freshness (Definition 4) would have to be modified. [1, Remark 7].

3. **The Game:** To begin, we must define what it means for an oracle $\Omega_U^{(n)}$ to be *fresh*, which itself requires that we define partnered sessions:

   Definition 3 (Partnered Oracles). *A pair of oracles $\Omega_A^{(n)}$ and $\Omega_B^{(m)}$ are called* partnered *if all of the following are true:*

(a) $A \in \mathcal{C}$ *and* $B \in \mathcal{S}$*, or* $A \in \mathcal{S}$ *and* $B \in \mathcal{C}$*.*
(b) $\Omega_A^{(n)}$ *and* $\Omega_B^{(m)}$ *have both accepted.*
(c) $\Omega_A^{(n)}$*'s peer is* B*, and* $\Omega_B^{(m)}$*'s peer is* A *(that is,* A *believes she is establishing a key with* B*, and* vice versa*).*
(d) $\Omega_A^{(n)}$ *and* $\Omega_B^{(m)}$ *have the same message transcript and session key.*
(e) No other oracle $\Omega_U^{(k)}$ *accepts with the same message transcript.*

Definition 4 (Fresh Oracle). *An oracle* $\Omega_U^{(n)}$ *is called* fresh *(or, to emphasize that we are explicitly considering forward secrecy in the weak corruption model,* weak forward secrecy fresh*) if none of the following is true:*
(a) `Reveal`(U, n) *has been issued.*
(b) `Reveal`(V, m) *has been issued, where* $\Omega_V^{(m)}$ *is the partner to* $\Omega_U^{(n)}$*.*
(c) `Corrupt`(V, i) *was issued for some* $V \in \mathcal{C} \sqcup \mathcal{S}$ *and* `Send`(U, n, M) *was issued for some* $M \in \mathcal{M}$*.*

Now, at any point during the protocol, $\mathcal{A}$ may make a `Test`(U, n) query. If $\Omega_U^{(n)}$ is not fresh or this is not the first `Test` query of the game, $\mathcal{A}$ loses the game; otherwise, the challenger answers the query appropriately. The game continues, and eventually $\mathcal{A}$ makes a guess b' at the value of b that the challenger chose in response to the `Test` query; the adversary wins if $b' = b$ and loses otherwise. We define the adversary's advantage to be

$$\mathrm{Adv}_{\Gamma}^{\Pi}(\mathcal{A}) = 2\mathbb{P}[\mathcal{A} \text{ wins the game } \Gamma \text{ with protocol } \Pi] - 1.$$

B Computational Assumptions of SIDH

In this section we present the SIDH and SSI problems [11, Problems 5.1–5.4], whose presumed hardness underlies the security of SIDH (and, by extension, our protocol).

Definition 5 (Computational Supersingular Isogeny Diffie-Hellman Problem (SIDH)). *Let* $\phi_A \colon E \to E_A$ *be an isogeny with kernel* $\langle P_A + n_A Q_A \rangle$ *for* $n_A \leftarrow \mathrm{Unif}(\mathbb{Z}/\ell_A^{e_A}\mathbb{Z})$*. Let* $\phi_B \colon E \to E_B$ *be an isogeny with kernel* $\langle P_B + n_B Q_B \rangle$ *for* $n_B \leftarrow \mathrm{Unif}(\mathbb{Z}/\ell_B^{e_B}\mathbb{Z})$*. The* Computational Supersingular Diffie-Hellman problem *(*SIDH*) is to find the* j*-invariant of* $E_{AB} = E/\langle P_A + n_A Q_A, P_B + n_B Q_B \rangle$ *given*

$$\big((E, P_A, Q_A, P_B, Q_B), (E_A, \phi_A(P_B), \phi_A(Q_B)), (E_B, \phi_B(P_A), \phi_B(Q_A))\big).$$

If $((E, P_A, Q_A, P_B, Q_B), (E_A, \phi_A(P_B), \phi_A(Q_B)), (E_B, \phi_B(P_A), \phi_B(Q_A)))$ is a valid SIDH instance under the notation of Definition 5, we write $E_{AB} = \mathrm{SIDH}((E, P_A, Q_A, P_B, Q_B), (E_A, \phi_A(P_B), \phi_A(Q_B)), (E_B, \phi_B(P_A), \phi_B(Q_A)))$.

When the global parameters and auxiliary points are clear from context, we abbreviate this as $E_{AB} = \mathrm{SIDH}(E_A, E_B)$. We also define variants of SIDH:

– For $E_A = E/\langle P_A + m_A Q_A \rangle$, say $\mathrm{SIDH}_A(E_A, E', X', Y') = E'/\langle X' + m_A Y' \rangle$

– For $E_B = E/\langle P_B + m_B Q_B\rangle$, say $\mathrm{SIDH}_B(E', E_B, X', Y') = E'/\langle X' + m_B Y'\rangle$

These arise in SIDH by considering messages which are not well-formed.

For an algorithm $\mathcal{A}$ which, given a valid SIDH instance, produces a list of candidate solutions to the instance, define its advantage as

$$\mathrm{Adv}_p^{\mathrm{SIDH}}(\mathcal{A}) = \mathbb{P}[\vartheta \leftarrow \mathrm{Unif}(\mathtt{Ins}_{\mathrm{SIDH}}(p)) : \mathcal{A}(\vartheta) = \mathrm{SIDH}(\vartheta)]$$

where $\mathtt{Ins}_{\mathrm{SIDH}}(p)$ is the set of all valid SIDH instances defined over $GF(p^2)$. Also define

$$\mathrm{Adv}_p^{\mathrm{SIDH}}(t) = \max\{\mathrm{Adv}_p^{\mathrm{SIDH}}(\mathcal{A}) : \mathcal{A} \text{ runs in time } t \text{ for all } \vartheta \in \mathtt{Ins}_{\mathrm{SIDH}}(p)\}.$$

Intuitively, this quantity measures the maximum probability of solving a randomly chosen SIDH instance in time t if you are allowed to make n guesses. The SIDH assumption is that for $t, n = \mathrm{poly}(\lambda)$, $\mathrm{Adv}_p^{\mathrm{SIDH}}(t, n) = \mathrm{negl}(\lambda)$.

As in Section 2.5, the asymmetry the isogeny computations requires us to consider there are really two variants of each of the following computational problem: one for $\ell_A^{e_A}$-isogenies, and the other for $\ell_B^{e_B}$-isogenies. For brevity we present the "Type A" problem and advantage and omit the analogous "Type B" variant.

Definition 6 (Supersingular Isogeny Problem–A (SSI_A)). *Let $\phi_A\colon E \to E_A$ be an isogeny with kernel $\langle P_A + n_A Q_A\rangle$ for $n_A \leftarrow \mathrm{Unif}(\mathbb{Z}/\ell_A^{e_A}\mathbb{Z})$. The* supersingular isogeny problem (type A) (SSI_A) *is, given E, E_A, $\phi_A(P_B)$, and $\phi_A(Q_B)$, to find a generator of* $\ker\,\phi_A$.

The SIDH problem reduces to both the Type A and type B variants of Problem 6 by noting that knowledge of $\ker\phi_A, \phi_B(P_A)$ and $\phi_B(Q_A)$ allows one to map E_B to E_{AB}, and the knowledge of $\ker\phi_B, \phi_A(P_B)$ and $\phi_A(Q_B)$ allows one to map E_A to E_{AB} similarly. Recent work due to Costache *et al.* [8, Theorem 3.2] reduces the SIDH problem to the more general isogeny-graph path-finding problem of [7] in a similar fashion.