Supersingular Isogeny Diffie–Hellman Authenticated Key Exchange

Atsushi FUJIOKA[1], Katsuyuki TAKASHIMA[2], Shintaro TERADA[3], and Kazuki YONEYAMA[3]

[1] Kanagawa University, Kanagawa, Japan
fujioka@kanagawa-u.ac.jp
[2] Mitsubishi Electric, Kanagawa, Japan
Takashima.Katsuyuki@aj.MitsubishiElectric.co.jp
[3] Ibaraki University, Ibaraki, Japan
{17nm713n,kazuki.yoneyama.sec}@vc.ibaraki.ac.jp

Abstract. We propose two authenticated key exchange protocols from supersingular isogenies. Our protocols are the first post-quantum one-round Diffie–Hellman type authenticated key exchange ones in the following points: one is secure under the quantum random oracle model and the other resists against maximum exposure where a non-trivial combination of secret keys is revealed. The security of the former and the latter is proven under isogeny versions of the decisional and gap Diffie–Hellman assumptions, respectively. We also propose a new approach for invalidating the Galbraith–Vercauteren-type attack for the gap problem.

Keywords: One-round authenticated key exchange · Supersingular isogeny decisional Diffie–Hellman assumption · Degree-insensitive supersingular isogeny gap Diffie–Hellman assumption · CK model · CK$^+$ model · Quantum adversary.

1 Introduction

All conventional cryptosystems from discrete logarithm and/or factorization intractability assumptions would be totally broken by the emergence of quantum computers, i.e., by Shor's algorithm [28]. In the post-quantum era, it is important to confirm whether classical cryptographic techniques are still secure against quantum adversaries. Recently, strong security notions and constructions against quantum computers have been intensively studied (e.g., [3, 36, 35, 10, 1]). Moreover, National Institute of Standards and Technology has initiated a process to standardize quantum-resistant public-key cryptographic algorithms [25], so, to study quantum-resistant cryptosystems is a hot research area.

Key establishing over insecure channels is one of important cryptographic techniques. In a key establishing protocol, two parties exchange some messages, and then, they can share a key. Recent researches on this have lead to *authenticated key exchange* (AKE). In the post-quantum era, it is preferable to have an AKE protocol secure based on a problem which resists against quantum adversaries. We then propose two quantum-resistant AKE schemes from a (relatively) new mathematical foundation, i.e., supersingular isogenies.

Supersingular Isogeny Diffie–Hellman (SIDH). Computing a sequence of isogenies of elliptic curves is a new cryptographic basic operation in some applications. For example, a cryptographic hash function from expander graphs, proposed in [6], consists of computing an isogeny sequence, which is based on the hardness of constructing an isogeny between two (randomly chosen) isogenous curves. Diffie–Hellman (DH) type key exchange protocols based on isogenies are given by Rostovtsev and Stolbunov [27] and De Feo et al. [11], which were considered as candidates for post-quantum public-key primitives.

Childs et al. [7] considered the isogeny computation problem for *ordinary* elliptic curves, and obtained a subexponential-time quantum algorithm. In contrast, the algorithm cannot be applied to the supersingular case (because of non-commutativity of endomorphism rings). Therefore, both applications above, i.e., hash function and key exchange, need to employ *supersingular* elliptic curves (and the graph consisting of them). In particular, *supersingular isogeny Diffie–Hellman* (SIDH) protocol proposed by De Feo et al. [11] has short public keys compared to other post-quantum candidates, and has been intensively studied for serving as a drop-in replacement to existing Internet protocols [9, 2, 8].

Very recently, Petit [26] proposed a mathematical attack for the security of SIDH, but also showed that the security is not affected by the attack if we use appropriate public parameters as is given in Sect. 3.

Authenticated Key Exchange. In an AKE protocol, two parties have own static public keys, exchange ephemeral public keys, and compute a session key based on the public keys and the related secret keys. AKE protocols achieve that honest parties can establish a session key, and any malicious party cannot guess the session key. The latter condition is formulated in an indistinguishability game.

Regarding to this security game, several models have been invented, and the Canetti–Krawczyk (CK) model was proposed to capture leakage of the session state [5]. After the proposal, several security requirements have been indicated such as *key compromise impersonation* (KCI), *weak perfect forward secrecy* (wPFS), and *maximal exposure attacks* (MEX) (refer to [22] for KCI, wPFS, and MEX). The CK model has been integrated with KCI, wPFS, and MEX to the CK^+ model [13].

Recently, several SIDH AKE protocols have been proposed [15, 24, 23, 34].

Galbraith proposed a one-round [4] protocol (SIDH TS2) in [15] based on the Unified Model DH protocol by Jeong, Katz, and Lee [19]. The protocol is CK-secure under a decisional problem in classical random oracle model (ROM).

Longa shows a two-round SIDH AKE protocol (AKE-SIDH-SIKE) which is CK^+-secure from a KEM scheme [24]. However, it is based on a generic construction known already.

[4] Galbraith claims that the protocol is one-round however the description shows that it is two-round as the responder generates the response after receiving the first message [15].

LeGrow, Jao, and Azarderakhsh defined a security model in which the adversary is allowed to make quantum queries, and proposed a *quantum* CK secure (qCK secure) protocol [23]. The protocol, we call it LJA, is secure in the quantum random oracle model (QROM) however it is two-round.

Xu et al. proposed a two-round protocol ($\mathrm{AKE}_{\text{SIDH-2}}$) in [34], and the protocol is CK^+-secure under a decisional problem in classical random oracle model (ROM).

It is worth to note here that all the existing SIDH AKE protocols shown above *only* achieve two-pass protocols except the SIDH TS2 protocol. In a one-round protocol, two parties can simultaneously exchange their ephemeral keys, while in a two-pass one, a party has to wait for the ephemeral key from the other party. Moreover, a one-round AKE protocol has several advantages of efficiency, e.g., each party can pre-compute ephemeral keys in advance.

Supersingular Isogeny Gap DH Problem. Traditional DH AKE protocols have been constructed from several forms of DH assumptions, i.e., computational, decisional and gap DH assumptions, for attaining various trade-offs between security and efficiency. Recently, Galbraith and Vercauteren [17] and Thormarker [31] independently proposed attacks, called *GV-type attack* in this paper, on the supersingular isogeny computational DH (SI-CDH) problem with access to *decision degree* oracle, which determines whether two supersingular curves are isogenous of some *specific degree* or not. While the attack can be extended to *some* form of SI version of gap DH (SI-GDH) problem, still, there exist possible approaches to formulate a *secure* form of SI-GDH problem (and assumption) for which the above attack is ineffective. Therefore, it is important to find and establish such *secure* SI-GDH assumptions to rescue (a wide range of) SIDH-based AKE schemes on the gap assumptions. (For surveys on SIDH-related computational problems, refer to [17, 32].)

Contributions. We propose two one-round authenticated key exchange protocols from supersingular isogenies: one is a protocol secure in the CK model with a quantum adversary under a supersingular isogeny version of the DDH assumption, and the other is a protocol secure in the CK^+ model with a classical adversary under a supersingular isogeny version of the gap DH assumption.

We call the latter assumption *degree-insensitive (di-)SI-GDH* assumption in which an adversary has access to a degree-insensitive SI-DDH oracle, and then cannot employ the GV-type attack for which degree distinction is crucial. We expect that the new assumption is of independent interest. Then, both protocols have several advantages of efficiency and wide applicability in practical situations as they retain a simple one-round Diffie–Hellman structure, and are realized in exchanging a single elliptic curve with an auxiliary smooth-order torsion basis, which can be efficiently compressed [2, 8]. We give a comparison table of the existing SIDH AKE protocols and our proposals in Table 1.

Table 1. Comparison of SIDH AKE protocols.

	assumption	model	action	proof
SIDH TS2 [15]	SI-CDH	CK	one-round [4]	ROM
AKE-SIDH-SIKE [24]	SI-DDH	CK$^+$	two-round	ROM
LJA [23]	SI-DDH	qCK	two-round	QROM
AKE$_{\text{SIDH-2}}$ [34]	SI-DDH	CK$^+$	two-round	ROM
SIDH UM	SI-DDH	CK	one-round	QROM
biclique SIDH	di-SI-GDH	CK$^+$	one-round	ROM

Notations. When A is a set, $y \in_R A$ denotes that y is uniformly selected from A. When A is a random variable, $y \leftarrow_R A$ denotes that y is randomly selected from A according to its distribution. We denote the finite field of order q by $\mathbb{F}_q$.

2 Security Models: CK-security and CK$^+$-security

This section outlines the CK and CK$^+$ security definitions for two-pass PKI-based authenticated key exchange protocols. Note that, in our *post-quantum* CK and CK$^+$ models, all parties are modeled by probabilistic polynomial-time (ppt) Turing machines while the adversary is modeled by a polynomial time quantum machine. For further CK and CK$^+$ details and explanations, see [22, 12]. It is worth to note here that the proposed protocols are one-round and thus, it is enough to describe the security model as for two-pass AKE because a two-pass model includes a one-round one.

We denote a party's identity $\hat{A}$, $\hat{B}$, $\hat{C}$, ..., where the ID space is **IDS**. A party honestly generates its own keys, static public and static secret ones, and the static public key is linked with the party's identity in some systems like PKI.[5] The maximum numbers of parties and sessions are polynomially bound in the security parameter.

We outline our models for a two-pass AKE protocol where parties, $\hat{A}$ and $\hat{B}$, exchange ephemeral public keys, X and Y, i.e., $\hat{A}$ sends X to $\hat{B}$ and $\hat{B}$ sends Y to $\hat{A}$, and thereafter derive a session key. The session key depends on the exchanged ephemeral keys, identifiers of the parties, the static keys, and the protocol instance that is used.

Keys. The public key owned by each party and its secret key are called *static public key* and *static secret key*, respectively. The one-time use session information exchanged in the protocol is called *ephemeral public key* as the information is generated from a temporary secret called *ephemeral secret key*.

Session. An invocation of a protocol is called a *session*. A session is activated via an incoming message of the forms $(\Pi, \mathcal{I}, \hat{A}, \hat{B})$ or $(\Pi, \mathcal{R}, \hat{A}, \hat{B}, Y)$, where

[5] Static public keys must be known to both parties in advance. They can be obtained by exchanging them before starting the protocol or by receiving them from a certificate authority. This situation is common for all PKI-based AKE protocols.

$\Pi \in \mathbf{PRS}$ is a protocol identifier in the protocol ID space, $\mathbf{PRS}$. If $\hat{A}$ is activated with $(\Pi, \mathcal{I}, \hat{A}, \hat{B})$, then $\hat{A}$ is the session *initiator*, otherwise it is the session *responder*. We say that $\hat{A}$ is the *owner* (resp. *peer*) of session $\mathtt{sid}$ if the third (resp. fourth) coordinate of $\mathtt{sid}$ is $\hat{A}$. After activation, session initiator $\hat{A}$ creates ephemeral public key X and a new session identified with $(\Pi, \mathcal{I}, \hat{A}, \hat{B}, X, \perp)$, and sends $(\Pi, \mathcal{R}, \hat{B}, \hat{A}, X)$ to the session responder $\hat{B}$, who then prepares ephemeral public key Y and a new session identified with $(\Pi, \mathcal{R}, \hat{B}, \hat{A}, X, Y)$, computes the session key and sends $(\Pi, \mathcal{I}, \hat{A}, \hat{B}, X, Y)$ to $\hat{A}$. Upon receiving $(\Pi, \mathcal{I}, \hat{A}, \hat{B}, X, Y)$, $\hat{A}$ updates the session identifier $(\Pi, \mathcal{I}, \hat{A}, \hat{B}, X, \perp)$ with $(\Pi, \mathcal{I}, \hat{A}, \hat{B}, X, Y)$ and computes a session key for that session. We say that a session is *completed* if its owner computes a session key.

If $\hat{A}$ is the initiator of a session, the session is identified via $\mathtt{sid} = (\Pi, \mathcal{I}, \hat{A}, \hat{B}, X, \perp)$ or $\mathtt{sid} = (\Pi, \mathcal{I}, \hat{A}, \hat{B}, X, Y)$. If $\hat{B}$ is the responder of a session, the session is identified via $\mathtt{sid} = (\Pi, \mathcal{R}, \hat{B}, \hat{A}, X, Y)$. The *matching session* of the session identified via $(\Pi, \mathcal{I}, \hat{A}, \hat{B}, X, Y)$ is a session with identifier $(\Pi, \mathcal{R}, \hat{B}, \hat{A}, X, Y)$ and vice versa.

Adversary. Adversary $\mathcal{M}$ is modeled as a probabilistic Turing machine that controls all communications including session activation. Activation is performed via a $\mathsf{Send}(\text{MESSAGE})$ query. The MESSAGE has one of the following forms: $(\Pi, \mathcal{I}, \hat{A}, \hat{B})$, $(\Pi, \mathcal{R}, \hat{A}, \hat{B}, X)$, or $(\Pi, \mathcal{I}, \hat{A}, \hat{B}, X, Y)$. Each party submits its responses to adversary $\mathcal{M}$, who decides the global delivery order.

The secret information of a party is not accessible to adversary $\mathcal{M}$; however, leakage of secret information is obtained via the following adversary queries.

- $\mathsf{SessionKeyReveal}(\mathtt{sid})$: $\mathcal{M}$ obtains the session key for the session with session identifier $\mathtt{sid}$, provided that the session is completed.
- $\mathsf{SessionStateReveal}(\mathtt{sid})$: $\mathcal{M}$ obtains the session state of the owner of session $\mathtt{sid}$ if the session is not completed (the session key is not established yet). The session state includes all ephemeral secret keys and intermediate computation results except for immediately erased information but does not include the static secret key.
- $\mathsf{Corrupt}(\hat{A})$: The query allows $\mathcal{M}$ to obtain all information of party $\hat{A}$. If a party, $\hat{A}$, is corrupted by a $\mathsf{Corrupt}(\hat{A})$ query issued by $\mathcal{M}$, then we call the party, $\hat{A}$, *dishonest*. If not, we call the party *honest*.

Definition 1 (Freshness). *Let* $\mathtt{sid}^*$ *be the session identifier of a completed session, owned by an honest party* $\hat{A}$ *with an honest peer* $\hat{B}$*. If the matching session exists, then let* $\overline{\mathtt{sid}^*}$ *be the session identifier of the matching session of* $\mathtt{sid}^*$*. Define* $\mathtt{sid}^*$ *to be* fresh *if none of the following conditions hold:*

- $\mathcal{M}$ *issues* $\mathsf{SessionKeyReveal}(\mathtt{sid}^*)$*, or* $\mathsf{SessionKeyReveal}(\overline{\mathtt{sid}^*})$ *if* $\overline{\mathtt{sid}^*}$ *exists.*
- $\overline{\mathtt{sid}^*}$ *exists and* $\mathcal{M}$ *makes either of the following queries*
  - $\mathsf{SessionStateReveal}(\mathtt{sid}^*)$ *or* $\mathsf{SessionStateReveal}(\overline{\mathtt{sid}^*})$*,*
- $\overline{\mathtt{sid}^*}$ *does not exist and* $\mathcal{M}$ *makes the following query*
  - $\mathsf{SessionStateReveal}(\mathtt{sid}^*)$.

Security Experiment. Initially, adversary $\mathcal{M}$ is given a set of honest parties, for whom $\mathcal{M}$ selects identifiers. Then the adversary makes any sequence of the queries described above. During the experiment, $\mathcal{M}$ makes a special query $\mathsf{Test}(\mathtt{sid}^*)$, where $\mathtt{sid}^*$ is the session identifier of a fresh session, and is given with equal probability either the session key held by $\mathtt{sid}^*$ or a random key; the query does not terminate the experiment. The experiment continues until $\mathcal{M}$ makes a guess whether the key is random or not. The adversary *wins* the game if the test session $\mathtt{sid}^*$ is still fresh and if the guess by $\mathcal{M}$ was correct. The advantage of quantum adversary $\mathcal{M}$ in the AKE experiment with AKE protocol Π is defined as

$$\mathbf{Adv}_{\Pi}^{\mathrm{AKE}}(\mathcal{M}) = \Pr[\mathcal{M} \text{ wins}] - \frac{1}{2}.$$

Definition 2 (Post-quantum CK security). *We say that an AKE protocol Π is post-quantum secure in the CK model if the following conditions hold:*

1. *If two honest parties complete matching sessions, then, except with negligible probability, they both compute the same session key.*
2. *For any polynomial-time quantum adversary $\mathcal{M}$, $\mathbf{Adv}_{\Pi}^{\mathrm{AKE}}(\mathcal{M})$ is negligible in security parameter λ for the test session $\mathtt{sid}^*$,*
   (a) *if $\overline{\mathtt{sid}^*}$ does not exist, or*
   (b) *if $\overline{\mathtt{sid}^*}$ exists, and the static secret key of the owner of $\mathtt{sid}^*$ and the static secret key of the owner of $\overline{\mathtt{sid}^*}$ are given to $\mathcal{M}$.*

Definition 3 (Post-quantum CK$^+$ security). *We say that an AKE protocol Π is post-quantum secure in the CK$^+$ model if the following conditions hold:*

1. *If two honest parties complete matching sessions, then, except with negligible probability, they both compute the same session key.*
2. *For any polynomial-time quantum adversary $\mathcal{M}$, $\mathbf{Adv}_{\Pi}^{\mathrm{AKE}}(\mathcal{M})$ is negligible in security parameter λ for the test session $\mathtt{sid}^*$,*
   (a) *if $\overline{\mathtt{sid}^*}$ does not exist, and the static secret key of the owner of $\mathtt{sid}^*$ is given to $\mathcal{M}$,*
   (b) *if $\overline{\mathtt{sid}^*}$ does not exist, and the ephemeral secret key of the owner of $\mathtt{sid}^*$ is given to $\mathcal{M}$,*
   (c) *if $\overline{\mathtt{sid}^*}$ exists, and the static secret key of the owner of $\mathtt{sid}^*$ and the static secret key of the owner of $\overline{\mathtt{sid}^*}$ are given to $\mathcal{M}$,*
   (d) *if $\overline{\mathtt{sid}^*}$ exists, and the ephemeral secret key of the owner of $\mathtt{sid}^*$ and the ephemeral secret key of the owner of $\overline{\mathtt{sid}^*}$ are given to $\mathcal{M}$,*
   (e) *if $\overline{\mathtt{sid}^*}$ exists, and the static secret key of the owner of $\mathtt{sid}^*$ and the ephemeral secret key of the owner of $\overline{\mathtt{sid}^*}$ are given to $\mathcal{M}$, or*
   (f) *if $\overline{\mathtt{sid}^*}$ exists, and the ephemeral secret key of the owner of $\mathtt{sid}^*$ and the static secret key of the owner of $\overline{\mathtt{sid}^*}$ are given to $\mathcal{M}$.*

The static and ephemeral public keys of our schemes include supersingular curves and points on them. We can test supersingularity of curves in polynomial time, e.g., [30]. We make an important remark: While Krawczyk mentions a strong adversary model where a corrupted party can choose to register any public

key of its choice at any point during the protocol as a variant of the CK($^+$) model in [22], we do not allow the re-registration of static public key (similar to the CK($^+$) model), and the initial public key is honestly generated and has been used until the end of the protocol. It is because that an active attack which Galbraith et al. [16] proposed for revealing static keys might be considered as an effective attack when we adopt the above flexible key re-registration.

3 Supersingular Isogeny Diffie–Hellman (SIDH)

We describe the SIDH protocol, whose implementation is investigated in detail in [9] and subsequently in [2, 21, 20, 4, 8]. The security is studied in [16, 26]. For making user secret keys short, we follow the description in the SIKE document [18], that is, the user key is given as just one scalar, e.g., $k_{\mathtt{A}} \in \mathbb{Z}/\ell_{\mathtt{A}}^{e_{\mathtt{A}}}\mathbb{Z}$. Refer to Appendix C for notations on elliptic curves.

3.1 Original (Concrete) Description of SIDH

For two small primes $\ell_{\mathtt{A}}, \ell_{\mathtt{B}}$ (e.g., $\ell_{\mathtt{A}} = 2, \ell_{\mathtt{B}} = 3$), we choose a large prime p such that $p \pm 1 = f \cdot \ell_{\mathtt{A}}^{e_{\mathtt{A}}} \ell_{\mathtt{B}}^{e_{\mathtt{B}}}$ for a small f and $\ell_{\mathtt{A}}^{e_{\mathtt{A}}} \approx \ell_{\mathtt{B}}^{e_{\mathtt{B}}} = 2^{\Theta(\lambda)}$, where λ is a security parameter. Then, we also choose a random supersingular elliptic curve E over $\mathbb{F}_{p^2}$ with $E(\mathbb{F}_{p^2}) \simeq (\mathbb{Z}/(p \pm 1)\mathbb{Z})^2 \supseteq (\mathbb{Z}/\ell_{\mathtt{A}}^{e_{\mathtt{A}}}\mathbb{Z})^2 \oplus (\mathbb{Z}/\ell_{\mathtt{B}}^{e_{\mathtt{B}}}\mathbb{Z})^2$. We use isogenies, $\phi_{\mathtt{A}}$ and $\phi_{\mathtt{B}}$, with kernels of orders, $\ell_{\mathtt{A}}^{e_{\mathtt{A}}}$ and $\ell_{\mathtt{B}}^{e_{\mathtt{B}}}$, respectively, and the following commutative diagram for the SIDH key exchange between Alice and Bob.

$$\begin{array}{ccc} E & \xrightarrow{\phi_{\mathtt{A}}} & E_{\mathtt{A}} = E/\langle R_{\mathtt{A}} \rangle \\ {\scriptstyle \phi_{\mathtt{B}}}\downarrow & & \downarrow{\scriptstyle \phi_{\mathtt{AB}}} \\ E_{\mathtt{B}} = E/\langle R_{\mathtt{B}} \rangle & \xrightarrow{\phi_{\mathtt{BA}}} & E/\langle R_{\mathtt{A}}, R_{\mathtt{B}} \rangle \end{array} \qquad \begin{array}{l} \text{for } \ker \phi_{\mathtt{A}} = \langle R_{\mathtt{A}} \rangle \subset E[\ell_{\mathtt{A}}^{e_{\mathtt{A}}}], \\ \quad \ker \phi_{\mathtt{B}} = \langle R_{\mathtt{B}} \rangle \subset E[\ell_{\mathtt{B}}^{e_{\mathtt{B}}}], \\ \quad \ker \phi_{\mathtt{BA}} = \langle \phi_{\mathtt{B}}(R_{\mathtt{A}}) \rangle \subset E_{\mathtt{B}}[\ell_{\mathtt{A}}^{e_{\mathtt{A}}}], \\ \quad \ker \phi_{\mathtt{AB}} = \langle \phi_{\mathtt{A}}(R_{\mathtt{B}}) \rangle \subset E_{\mathtt{A}}[\ell_{\mathtt{B}}^{e_{\mathtt{B}}}]. \end{array}$$

Below we first choose generators $P_{\mathtt{A}}, Q_{\mathtt{A}}, P_{\mathtt{B}}, Q_{\mathtt{B}}$ such that $E[\ell_{\mathtt{A}}^{e_{\mathtt{A}}}] = \langle P_{\mathtt{A}}, Q_{\mathtt{A}} \rangle$, $E[\ell_{\mathtt{B}}^{e_{\mathtt{B}}}] = \langle P_{\mathtt{B}}, Q_{\mathtt{B}} \rangle$ and then set the random curve $E/\mathbb{F}_{p^2}$ and the above generators as public parameters, i.e., we define the generator as $\mathsf{pk}^{\mathsf{sidh}} = (\mathfrak{g} = (E;\ P_{\mathtt{A}},\ Q_{\mathtt{A}},\ P_{\mathtt{B}},\ Q_{\mathtt{B}}),\ \mathfrak{e} = (\ell_{\mathtt{A}}, \ell_{\mathtt{B}}, e_{\mathtt{A}}, e_{\mathtt{B}})) \leftarrow_R \mathsf{Gen}^{\mathsf{sidh}}(1^\lambda)$. Secret-key spaces for Alice and Bob are given as $SK_{\mathtt{A}} = \mathbb{Z}/\ell_{\mathtt{A}}^{e_{\mathtt{A}}}\mathbb{Z}$ and $SK_{\mathtt{B}} = \mathbb{Z}/\ell_{\mathtt{B}}^{e_{\mathtt{B}}}\mathbb{Z}$, respectively. DH-type key exchange is given as below (Fig. 1). Here, since $\langle \phi_{\mathtt{B}}(P_{\mathtt{A}}) + k_{\mathtt{A}}\, \phi_{\mathtt{B}}(Q_{\mathtt{A}}) \rangle = \langle \phi_{\mathtt{B}}(R_{\mathtt{A}}) \rangle = \ker \phi_{\mathtt{BA}}$ and $\langle \phi_{\mathtt{A}}(P_{\mathtt{B}}) + k_{\mathtt{B}}\, \phi_{\mathtt{A}}(Q_{\mathtt{B}}) \rangle = \langle \phi_{\mathtt{A}}(R_{\mathtt{B}}) \rangle = \ker \phi_{\mathtt{AB}}$ hold, we have the equality of the j-invariants $K_{\mathtt{Alice}} = j(E_{\mathtt{B}}/\ker \phi_{\mathtt{BA}}) = j(E/\langle R_{\mathtt{A}}, R_{\mathtt{B}} \rangle) = j(E_{\mathtt{A}}/\ker \phi_{\mathtt{AB}}) = K_{\mathtt{Bob}}$, and $K = K_{\mathtt{Alice}} = K_{\mathtt{Bob}}$ is a shared key. Alice's output includes $\phi_{\mathtt{A}}(P_{\mathtt{B}})$ and $\phi_{\mathtt{A}}(Q_{\mathtt{B}})$ as well as $E_{\mathtt{A}}$, and the security is based on the hardness of isogeny problem with the auxiliary inputs.

3.2 Crypto-friendly Description of SIDH

We prepare an alternative crypto-friendly description of SIDH for a simple presentation of our proposed AKE.

Alice		**Bob**
$k_{\mathtt{A}} \in_R SK_{\mathtt{A}}$:		$k_{\mathtt{B}} \in_R SK_{\mathtt{B}}$:
Alice′s secret key,	$\xrightarrow{E_{\mathtt{A}},\, \phi_{\mathtt{A}}(P_{\mathtt{B}}),\, \phi_{\mathtt{A}}(Q_{\mathtt{B}})}$	Bob′s secret key,
$R_{\mathtt{A}} = P_{\mathtt{A}} + k_{\mathtt{A}} Q_{\mathtt{A}}$,	$\xleftarrow{E_{\mathtt{B}},\, \phi_{\mathtt{B}}(P_{\mathtt{A}}),\, \phi_{\mathtt{B}}(Q_{\mathtt{A}})}$	$R_{\mathtt{B}} = P_{\mathtt{B}} + k_{\mathtt{B}} Q_{\mathtt{B}}$,
$\phi_{\mathtt{A}} : E \to E_{\mathtt{A}} = E/\langle R_{\mathtt{A}} \rangle$,		$\phi_{\mathtt{B}} : E \to E_{\mathtt{B}} = E/\langle R_{\mathtt{B}} \rangle$,
$R_{\mathtt{BA}} = \phi_{\mathtt{B}}(P_{\mathtt{A}}) + k_{\mathtt{A}}\, \phi_{\mathtt{B}}(Q_{\mathtt{A}})$,		$R_{\mathtt{AB}} = \phi_{\mathtt{A}}(P_{\mathtt{B}}) + k_{\mathtt{B}}\, \phi_{\mathtt{A}}(Q_{\mathtt{B}})$,
$K_{\mathtt{Alice}} = j(E_{\mathtt{B}}/\langle R_{\mathtt{BA}} \rangle)$.		$K_{\mathtt{Bob}} = j(E_{\mathtt{A}}/\langle R_{\mathtt{AB}} \rangle)$.

Fig. 1. Outline of SIDH Protocol (Original Description).

We set

$$\mathfrak{g} = (E;\ P_{\mathtt{A}},\ Q_{\mathtt{A}},\ P_{\mathtt{B}},\ Q_{\mathtt{B}}),\ \mathfrak{a} = k_{\mathtt{A}},\ \text{and}\ \mathfrak{b} = k_{\mathtt{B}}.$$

Let the sets of supersingular curves and those with an auxiliary torsion basis be

$$\begin{aligned}
SSEC_p &= \{\, \text{supersingular elliptic curve } E \text{ over } \mathbb{F}_{p^2} \\
&\qquad \text{with } E(\mathbb{F}_{p^2}) \simeq (\mathbb{Z}/(p \pm 1)\mathbb{Z})^2 \supseteq (\mathbb{Z}/\ell_{\mathtt{A}}^{e_{\mathtt{A}}}\mathbb{Z})^2 \oplus (\mathbb{Z}/\ell_{\mathtt{B}}^{e_{\mathtt{B}}}\mathbb{Z})^2 \},\\
SSEC_{p,\mathtt{A}} &= \{\, (E;\ P'_{\mathtt{B}},\ Q'_{\mathtt{B}}) \,|\, E \in SSEC_p,\ (P'_{\mathtt{B}},\ Q'_{\mathtt{B}}) : \text{basis of } E[\ell_{\mathtt{B}}^{e_{\mathtt{B}}}] \},\\
SSEC_{p,\mathtt{B}} &= \{\, (E;\ P'_{\mathtt{A}},\ Q'_{\mathtt{A}}) \,|\, E \in SSEC_p,\ (P'_{\mathtt{A}},\ Q'_{\mathtt{A}}) : \text{basis of } E[\ell_{\mathtt{A}}^{e_{\mathtt{A}}}] \}.
\end{aligned}$$

Thus, SIDH public keys of $\mathtt{A}$ and $\mathtt{B}$ are given elements of $SSEC_{p,\mathtt{A}}$ and $SSEC_{p,\mathtt{B}}$, respectively. Then, we define

$$\begin{aligned}
&\mathfrak{g}^{\mathfrak{a}} = (E_{\mathtt{A}};\ \phi_{\mathtt{A}}(P_{\mathtt{B}}),\ \phi_{\mathtt{A}}(Q_{\mathtt{B}})) \in SSEC_{p,\mathtt{A}},\\
&\quad \text{where } R_{\mathtt{A}} = P_{\mathtt{A}} + k_{\mathtt{A}} Q_{\mathtt{A}},\ \phi_{\mathtt{A}} : E \to E_{\mathtt{A}} = E/\langle R_{\mathtt{A}} \rangle,\\
&\mathfrak{g}^{\mathfrak{b}} = (E_{\mathtt{B}};\ \phi_{\mathtt{B}}(P_{\mathtt{A}}),\ \phi_{\mathtt{B}}(Q_{\mathtt{A}})) \in SSEC_{p,\mathtt{B}},\\
&\quad \text{where } R_{\mathtt{B}} = P_{\mathtt{B}} + k_{\mathtt{B}} Q_{\mathtt{B}},\ \phi_{\mathtt{B}} : E \to E_{\mathtt{B}} = E/\langle R_{\mathtt{B}} \rangle,\\
&\left(\mathfrak{g}^{\mathfrak{b}}\right)^{\mathfrak{a}} = j(E_{\mathtt{BA}}),\\
&\quad \text{where } R_{\mathtt{BA}} = \phi_{\mathtt{B}}(P_{\mathtt{A}}) + k_{\mathtt{A}} \phi_{\mathtt{B}}(Q_{\mathtt{A}}),\ \phi_{\mathtt{BA}} : E_{\mathtt{B}} \to E_{\mathtt{BA}} = E_{\mathtt{B}}/\langle R_{\mathtt{BA}} \rangle,\\
&(\mathfrak{g}^{\mathfrak{a}})^{\mathfrak{b}} = j(E_{\mathtt{AB}}),\\
&\quad \text{where } R_{\mathtt{AB}} = \phi_{\mathtt{A}}(P_{\mathtt{B}}) + k_{\mathtt{B}} \phi_{\mathtt{A}}(Q_{\mathtt{B}}),\ \phi_{\mathtt{AB}} : E_{\mathtt{A}} \to E_{\mathtt{AB}} = E_{\mathtt{A}}/\langle R_{\mathtt{AB}} \rangle.
\end{aligned}$$

We describe SIDH using this notation below (Fig. 2). Public parameters are $\mathfrak{g} = (E;\ P_{\mathtt{A}},\ Q_{\mathtt{A}},\ P_{\mathtt{B}},\ Q_{\mathtt{B}})$ and $\mathfrak{e} = (\ell_{\mathtt{A}}, \ell_{\mathtt{B}}, e_{\mathtt{A}}, e_{\mathtt{B}})$. Here, shared secret is given as $K_{\mathtt{Alice}} = \left(\mathfrak{g}^{\mathfrak{b}}\right)^{\mathfrak{a}} = (\mathfrak{g}^{\mathfrak{a}})^{\mathfrak{b}} = K_{\mathtt{Bob}}$, which shows correctness of the SIDH protocol.

4 Post-Quantum Assumptions from SIDH

We define SI-CDH, SI-DDH, ds- and di-SI-GDH assumptions against quantum adversaries based on the notation in Sect. 3.2. The SI-DDH assumption is needed for indistinguishability security of SIDH shared keys. Moreover, all of the following assumptions excluding ds-SI-GDH (see Prop. 1) are considered reasonable at present.

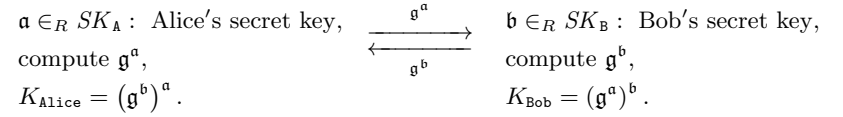

Fig. 2. Outline of SIDH Protocol (Crypto-friendly Description).

Definition 4 (SI-CDH Assumption). *Let $\mathcal{S}$ be a quantum machine adversary. For $\mathsf{pk}^{\mathsf{sidh}} = (\mathfrak{g} = (E;\ P_{\mathtt{A}},\ Q_{\mathtt{A}},\ P_{\mathtt{B}},\ Q_{\mathtt{B}}),\ \mathfrak{e} = (\ell_{\mathtt{A}}, \ell_{\mathtt{B}}, e_{\mathtt{A}}, e_{\mathtt{B}})) \leftarrow_R \mathsf{Gen}^{\mathsf{sidh}}(1^\lambda)$ and $\mathfrak{a} \in_R SK_{\mathtt{A}},\ \mathfrak{b} \in_R SK_{\mathtt{B}}$, $\mathcal{S}$ receives $(\ \mathsf{pk}^{\mathsf{sidh}},\ \mathfrak{g}^{\mathfrak{a}},\ \mathfrak{g}^{\mathfrak{b}})$, and $\mathcal{S}$ outputs $\mathfrak{h} \in \mathbb{F}_{p^2}$. If $\mathfrak{h} = (\mathfrak{g}^{\mathfrak{a}})^{\mathfrak{b}}\ (= \left(\mathfrak{g}^{\mathfrak{b}}\right)^{\mathfrak{a}})$, $\mathcal{S}$ wins. We define the advantage of $\mathcal{S}$ for the SI-CDH problem as $\mathbf{Adv}^{\text{SI-CDH}}_{\mathfrak{g},\mathfrak{e}}(\mathcal{S}) = \Pr[\mathcal{S}\ wins]$. The SI-CDH assumption is: For any polynomial-time quantum machine adversary $\mathcal{S}$, the advantage of $\mathcal{S}$ for the SI-CDH problem is negligible in security parameter λ.*

Definition 5 (SI-DDH Assumption). *Let $\mathcal{S}$ be a quantum machine adversary. For $\mathsf{pk}^{\mathsf{sidh}} = (\mathfrak{g} = (E;\ P_{\mathtt{A}},\ Q_{\mathtt{A}},\ P_{\mathtt{B}},\ Q_{\mathtt{B}}),\ \mathfrak{e} = (\ell_{\mathtt{A}}, \ell_{\mathtt{B}}, e_{\mathtt{A}}, e_{\mathtt{B}})) \leftarrow_R \mathsf{Gen}^{\mathsf{sidh}}(1^\lambda)$ and $\mathfrak{a}, \mathfrak{r} \in_R SK_{\mathtt{A}},\ \mathfrak{b}, \mathfrak{s} \in_R SK_{\mathtt{B}}$, $\mathcal{S}$ receives $\mathcal{X}_b$ for $b \in_R \{0,1\}$, that is defined by*

$$\mathcal{X}_0 = (\ \mathsf{pk}^{\mathsf{sidh}},\ \mathfrak{g}^{\mathfrak{a}},\ \mathfrak{g}^{\mathfrak{b}},\ (\mathfrak{g}^{\mathfrak{a}})^{\mathfrak{b}}\ ) \ \text{ and } \ \mathcal{X}_1 = (\ \mathsf{pk}^{\mathsf{sidh}},\ \mathfrak{g}^{\mathfrak{a}},\ \mathfrak{g}^{\mathfrak{b}},\ (\mathfrak{g}^{\mathfrak{r}})^{\mathfrak{s}}\ ),$$

$\mathcal{S}$ outputs a guess bit b'. If $b = b'$, $\mathcal{S}$ wins. We define the advantage of $\mathcal{S}$ for the SI-DDH problem as $\mathbf{Adv}^{\text{SI-DDH}}_{\mathfrak{g},\mathfrak{e}}(\mathcal{S}) = \Pr[\mathcal{S}\ wins] - 1/2$. The SI-DDH assumption is: For any polynomial-time quantum machine adversary $\mathcal{S}$, the advantage of $\mathcal{S}$ for the SI-DDH problem is negligible in security parameter λ.

Definition 6 (ds- and di-SI-GDH Assumption). *Let $\mathcal{S}$ be a quantum machine adversary. For $\mathsf{pk}^{\mathsf{sidh}} = (\mathfrak{g} = (E;\ P_{\mathtt{A}},\ Q_{\mathtt{A}},\ P_{\mathtt{B}},\ Q_{\mathtt{B}}),\ \mathfrak{e} = (\ell_{\mathtt{A}}, \ell_{\mathtt{B}}, e_{\mathtt{A}}, e_{\mathtt{B}})) \leftarrow_R \mathsf{Gen}^{\mathsf{sidh}}(1^\lambda)$ and $\mathfrak{a} \in_R SK_{\mathtt{A}},\ \mathfrak{b} \in_R SK_{\mathtt{B}}$, $\mathcal{S}$ receives $(\mathsf{pk}^{\mathsf{sidh}},\ \mathfrak{g},\ \mathfrak{g}^{\mathfrak{a}},\ \mathfrak{g}^{\mathfrak{b}})$, and $\mathcal{S}$ access SI-DDH oracle for any input $\mathcal{X} = (\mathsf{pk}^{\mathsf{sidh}}, (E'_{\mathtt{A}}; P'_{\mathtt{AB}}, Q'_{\mathtt{AB}}),\ (E'_{\mathtt{B}}; P'_{\mathtt{BA}}, Q'_{\mathtt{BA}}),\ \mathfrak{h}')$ where $P'_{\mathtt{AB}}, Q'_{\mathtt{AB}}$ (resp. $P'_{\mathtt{BA}}, Q'_{\mathtt{BA}}$) are points in $E'_{\mathtt{A}}(\mathbb{F}_{p^2})$ (resp. $E'_{\mathtt{B}}(\mathbb{F}_{p^2})$) and $\mathfrak{h}' \in \mathbb{F}_{p^2}$, and then outputs $\mathfrak{h} \in \mathbb{F}_{p^2}$. If $\mathfrak{h} = (\mathfrak{g}^{\mathfrak{a}})^{\mathfrak{b}}\ (= \left(\mathfrak{g}^{\mathfrak{b}}\right)^{\mathfrak{a}})$, $\mathcal{S}$ wins. According to the behavior of SI-DDH oracle, we have two types of SI-GDH problem, i.e.,*

- **degree-sensitive SI-GDH (ds-SI-GDH) problem** *The ds-SI-DDH oracle answers true if there exist a supersingular elliptic curve $E'_{\mathtt{AB}}$ and isogenies $(\phi'_{\mathtt{A}},\ \phi'_{\mathtt{B}},\ \phi'_{\mathtt{AB}},\ \phi'_{\mathtt{BA}})$ among $E, E'_{\mathtt{A}}, E'_{\mathtt{B}}, E'_{\mathtt{AB}}$ which form a commutative diagram as in Fig. 3 such that*
  - *degree $d'_{\mathtt{A}}$ of $\phi'_{\mathtt{A}}$ (and $\phi'_{\mathtt{BA}}$) is equal to $\ell_{\mathtt{A}}^{e_{\mathtt{A}}}$ and degree $d'_{\mathtt{B}}$ of $\phi'_{\mathtt{B}}$ (and $\phi'_{\mathtt{AB}}$) is equal to $\ell_{\mathtt{B}}^{e_{\mathtt{B}}}$ and*
  - *$P'_{\mathtt{AB}} = \phi'_{\mathtt{A}}(P_{\mathtt{B}})$, $Q'_{\mathtt{AB}} = \phi'_{\mathtt{A}}(Q_{\mathtt{B}})$ and $P'_{\mathtt{BA}} = \phi'_{\mathtt{B}}(P_{\mathtt{A}})$, $Q'_{\mathtt{BA}} = \phi'_{\mathtt{B}}(Q_{\mathtt{A}})$ where points $(P_{\mathtt{A}}, Q_{\mathtt{A}}, P_{\mathtt{B}}, Q_{\mathtt{B}})$ are given in public key $\mathsf{pk}^{\mathsf{sidh}}$, and $\mathfrak{h}' = j(E'_{\mathtt{AB}})$,*

  and false otherwise. We call this case degree-sensitive *SI-GDH (ds-SI-GDH) problem.*

– **degree-insensitive SI-GDH (di-SI-GDH) problem** *The di-SI-DDH oracle answers true if there exist a supersingular elliptic curve $E'_{\mathtt{AB}}$ and isogenies $(\phi'_{\mathtt{A}},\ \phi'_{\mathtt{B}},\ \phi'_{\mathtt{AB}},\ \phi'_{\mathtt{BA}})$ among $E, E'_{\mathtt{A}}, E'_{\mathtt{B}}, E'_{\mathtt{AB}}$ which form a commutative diagram as in Fig. 3 such that*
  - *degree $d'_{\mathtt{A}}$ of $\phi'_{\mathtt{A}}$ (and $\phi'_{\mathtt{BA}}$) is a power of $\ell_{\mathtt{A}}$ and degree $d'_{\mathtt{B}}$ of $\phi'_{\mathtt{B}}$ (and $\phi'_{\mathtt{AB}}$) is a power of $\ell_{\mathtt{B}}$ and*
  - *$P'_{\mathtt{AB}} = \phi'_{\mathtt{A}}(P_{\mathtt{B}}),\ Q'_{\mathtt{AB}} = \phi'_{\mathtt{A}}(Q_{\mathtt{B}})$ and $P'_{\mathtt{BA}} = \phi'_{\mathtt{B}}(P_{\mathtt{A}}),\ Q'_{\mathtt{BA}} = \phi'_{\mathtt{B}}(Q_{\mathtt{A}})$ where points $(P_{\mathtt{A}}, Q_{\mathtt{A}}, P_{\mathtt{B}}, Q_{\mathtt{B}})$ are given in public key $\mathsf{pk}^{\mathsf{sidh}}$, and $\mathfrak{h}' = j(E'_{\mathtt{AB}})$,*

  and false otherwise. We call this case degree-insensitive *SI-GDH (di-SI-GDH) problem.*

We define the advantage of adversary $\mathcal{S}$ for the ds–SI-GDH and di-SI-GDH problems as $\mathbf{Adv}^{\text{ds-SI-GDH}}_{\mathfrak{g},\mathfrak{e}}(\mathcal{S}) = \Pr[\mathcal{S}\ wins]$ and $\mathbf{Adv}^{\text{di-SI-GDH}}_{\mathfrak{g},\mathfrak{e}}(\mathcal{S}) = \Pr[\mathcal{S}\ wins]$, respectively. The ds-SI-GDH (resp. di-SI-GDH) assumption is: For any polynomial-time quantum machine adversary $\mathcal{S}$, the advantage of $\mathcal{S}$ for the ds-SI-GDH (resp. di-SI-GDH) problem is negligible in security parameter λ.

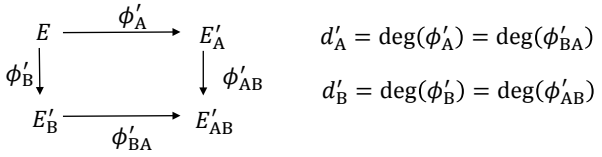

Fig. 3. Commutative diagram for true instances of SI-DDH oracles, in which it holds that $\ker(\phi'_{\mathtt{BA}}) = \phi'_{\mathtt{B}}(\ker(\phi'_{\mathtt{A}}))$ and $\ker(\phi'_{\mathtt{AB}}) = \phi'_{\mathtt{A}}(\ker(\phi'_{\mathtt{B}}))$.

Proposition 1 (adapted from [17]). *The ds-SI-GDH assumption does not hold, i.e., there exists a ppt adversary against the ds-SI-GDH problem.*

proof sketch. Very recently, Galbraith and Vercauteren proposed an attack on the SI-CDH problem with access to the decision degree (DD) oracle [17], which determines whether two supersingular curves are isogenous of some specific degree or not. As a basic building block, first, we describe an attack on the SI-CDH problem using the DD oracle. The input of the problem is $(\mathsf{pk}^{\mathsf{sidh}} = (\mathfrak{g} = (E;\ P_{\mathtt{A}},\ Q_{\mathtt{A}},\ P_{\mathtt{B}},\ Q_{\mathtt{B}}),\ \mathfrak{e} = (\ell_{\mathtt{A}}, \ell_{\mathtt{B}}, e_{\mathtt{A}}, e_{\mathtt{B}})), E_{\mathtt{A}}, P_{\mathtt{AB}}, Q_{\mathtt{AB}})$, where $\phi_{\mathtt{A}} : E \to E_{\mathtt{A}}$ is an $\ell_{\mathtt{A}}^{e_{\mathtt{A}}}$-isogeny, $P_{\mathtt{AB}} = \phi_{\mathtt{A}}(P_{\mathtt{B}})$, and $Q_{\mathtt{AB}} = \phi_{\mathtt{A}}(Q_{\mathtt{B}})$. The goal of the adversary $\mathcal{S}$ is to reveal $\phi_{\mathtt{A}}$. For that, $\mathcal{S}$ calculates integer u such that $u \cdot \ell_{\mathtt{A}} \equiv 1 \pmod{\ell_{\mathtt{B}}}$, and then one $\ell_{\mathtt{A}}$-isogeny $\psi : E_{\mathtt{A}} \to E'$. $\mathcal{S}$ send

$$(\tilde{\mathsf{pk}}^{\mathsf{sidh}} = (\mathfrak{g},\ \tilde{\mathfrak{e}} = (\ell_{\mathtt{A}}, \ell_{\mathtt{B}}, e_{\mathtt{A}} - 1, e_{\mathtt{B}}), E', u \cdot \psi(P_{\mathtt{AB}}), u \cdot \psi(Q_{\mathtt{AB}}))$$

to the DD oracle. Here, we note that the exponent $e_{\mathtt{A}} - 1$ is used instead of $e_{\mathtt{A}}$ for the implicitly defined $\ell_{\mathtt{A}}$-power isogeny. That is, the oracle distinguishes the degree (or length) of the isogeny, in other words, whether E' is $\ell_{\mathtt{A}}^{e_{\mathtt{A}}-1}$-isogenous to E or $\ell_{\mathtt{A}}^{e_{\mathtt{A}}+1}$-isogenous to E. See the left hand side of Fig. 4. Then, the adversary reveals all the isogeny by repeating this $\ell_{\mathtt{A}}$-backtracking decision.

Next, we extend the above strategy to solve the ds-SI-GDH problem. Namely, an ds-SI-GDH adversary obtains an input $(\mathsf{pk}^{\mathsf{sidh}} = (\mathfrak{g} = (E;\ P_{\mathtt{A}},\ Q_{\mathtt{A}},\ P_{\mathtt{B}},\ Q_{\mathtt{B}}),\ \mathfrak{e} = (\ell_{\mathtt{A}}, \ell_{\mathtt{B}}, e_{\mathtt{A}}, e_{\mathtt{B}})), E_{\mathtt{A}}, P_{\mathtt{AB}}, Q_{\mathtt{AB}}, \ldots)$, where $\phi_{\mathtt{A}} : E \to E_{\mathtt{A}}$ is an $\ell_{\mathtt{A}}^{e_{\mathtt{A}}}$-isogeny, $P_{\mathtt{AB}} = \phi_{\mathtt{A}}(P_{\mathtt{B}})$, and $Q_{\mathtt{AB}} = \phi_{\mathtt{A}}(Q_{\mathtt{B}})$. The goal of the adversary $\mathcal{S}$ is to reveal $\phi_{\mathtt{A}}$. For that, $\mathcal{S}$ calculates one $\ell_{\mathtt{A}}$-isogeny $\psi : E_{\mathtt{A}} \to E'$ as before. Moreover, $\mathcal{S}$ calculates degree $\ell_{\mathtt{B}}^{e_{\mathtt{B}}}$-isogenies $E \to E'_{\mathtt{B}}$ and $E' \to E'_{\mathtt{AB}}$ that makes commutative SIDH diagram $(E, E', E'_{\mathtt{B}}, E'_{\mathtt{AB}})$. Then, $\mathcal{S}$ send

$$(\tilde{\mathsf{pk}}^{\mathsf{sidh}} = (\mathfrak{g},\ \tilde{\mathfrak{e}} = (\ell_{\mathtt{A}}, \ell_{\mathtt{B}}, e_{\mathtt{A}} - 1, e_{\mathtt{B}}), E', E'_{\mathtt{B}}, \ldots, j(E'_{\mathtt{AB}}))$$

to the ds-SI-DDH oracle and determine whether ψ is a backtracking step in $\phi_{\mathtt{A}}$ or not. See the right hand side of Fig. 4. From here on, repeating this procedure, $\mathcal{S}$ can reveal $\phi_{\mathtt{A}}$. Also, $\mathcal{S}$ can compute $E_{\mathtt{AB}}$ by using $E_{\mathtt{B}}$ and $\phi_{\mathtt{A}}$, which solves the ds-SI-GDH problem. □

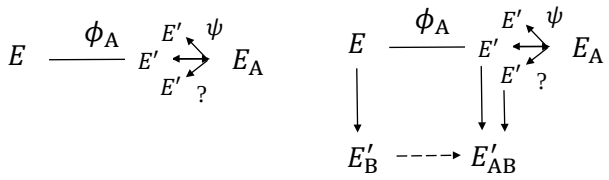

Fig. 4. Diagrams for the GV-type attack. The right (resp. left) hand side shows the strategy for the ds-SI-GDH problem (resp. the SI-CDH problem with access to the DD oracle). The attacker distinguishes which one of the $e_{\mathtt{A}} + 1$ red $\ell_{\mathtt{A}}$-isogenies is backtracking by using the ds-SI-DDH (resp. the DD) oracle.

As described in the above proof, to distinguish the degree of isogeny (or distance between two elliptic curves in the $\ell_{\mathtt{A}}$-isogeny graph) is crucial for the GV-type attack. Since the ability for the distinction is given by the ds-SI-DDH oracle, the GV-type attack adversaries have *no advantages* in the di-SI-GDH problem. Therefore, in contrast to the ds-SI-GDH problem, we may assume that the di-SI-GDH problem cannot be solved by any efficient adversaries, and can be used for the basis of the security of our biclique scheme.

Note that auxiliary points $\phi'_{\mathtt{A}}(P_{\mathtt{B}}), \phi'_{\mathtt{A}}(Q_{\mathtt{B}}), \phi'_{\mathtt{B}}(P_{\mathtt{A}}), \phi'_{\mathtt{B}}(Q_{\mathtt{A}})$ in true instance $\mathcal{X}$ for di-SI-DDH oracle impose some restrictions on implicitly defined isogenies $\phi'_{\mathtt{A}}, \phi'_{\mathtt{B}}$ (and $\phi'_{\mathtt{AB}}, \phi'_{\mathtt{BA}}$) used in Fig. 3. However, since degrees $d'_{\mathtt{A}}$ and $d'_{\mathtt{B}}$ of $\phi'_{\mathtt{A}}$ and $\phi'_{\mathtt{B}}$ can be chosen as *any* powers of $\ell_{\mathtt{A}}$ and $\ell_{\mathtt{B}}$ respectively, a wide range of tuples $(E'_{\mathtt{A}}, E'_{\mathtt{B}}, E'_{\mathtt{AB}})$ can be accepted for forming the commutative diagram in Fig. 3. Therefore, as an extreme possible case, *any* tuple of supersingular elliptic curves $(E'_{\mathtt{A}}, E'_{\mathtt{B}}, E'_{\mathtt{AB}})$ *might* form the commutative diagram in Fig. 3, that is, any tuple of such curves would be true instances in the hypothetical case. We cannot exclude such possibility from our present knowledge of the di-SI-GDH problem. A satisfiable analysis of the di-SI-GDH problem seems to need more understanding of the Ramanujan graph of ℓ-isogenies of supersingular curves.

Lemma 3.2 and Theorem 3.3 in [32] also show some interesting connection between computational and decisional SIDH problems. However, we notice that answers of all the oracles $(O_{E,1})_{\ell^e}, (O_{E,2})_{\ell^e}$ and $(O_{E,3})_{\ell^e}$ (for $\ell^e = \ell_1^{e_1}$ or $\ell_2^{e_2}$) are related to isogenies of degrees dividing ℓ^e, which is defined by public parameters. In particular, all the isogeny degrees have smaller or equal than ℓ^e. Our di-SI-GDH problem is related to unbounded degrees which are just a power of ℓ. Thus, Lemma 3.2 and Theorem 3.3 in [32] are now unrelated with our situation, but, we think seeking relationships between the di-SI-GDH problem and the results in [32] is an interesting research direction.

5 Proposed SIDH UM Protocol

In this section, we propose the SIDH UM protocol, where it can be proved in the quantum random oracle model under the SI-DDH assumption.

Before describing the protocol, we explain that each party needs to have two static public keys. The public parameter, $\mathfrak{g}$, contains two parameters, $(P_1,\ Q_1)$ and (P_2, Q_2). A party has a key on $(P_1,\ Q_1)$ and the other key on (P_2, Q_2). Then, $(P_1,\ Q_1)$ is used to generate the ephemeral public key of the initiator and $(P_2,\ Q_2)$ is used to generate the ephemeral public key of the responder. When the role is exchanged, each party uses the other static key which is not used before.

This double construction in public parameter and static public keys gives resistance to reflection attacks. To the best of our knowledge, the previous researches of key exchange on supersingular isogenies have lacked this consideration.

5.1 Useful Techniques for Quantum Random Oracle Model

A problem on security proofs in the quantum random oracle model is how to generate random values for exponentially many positions in order to simulate outputs of the hash function. For a hash function $H : Dom \to Rng$, in the quantum random oracle model, the adversary poses a superposition $|\phi\rangle = \Sigma\alpha_x|x\rangle$ and the oracle returns $\Sigma\alpha_x|H(x)\rangle$. If Rng is large for a quantum polynomial-time simulator, it is difficult to generate all random output values of H to compute $\Sigma\alpha_x|H(x)\rangle$. Zhandry [36] showed a solution with the notion of k-wise independent function.

A weight assignment on a set $\mathcal{X}$ is a function $D : \mathcal{X} \to \mathbb{R}$ such that $\Sigma_{x\in\mathcal{X}} D(x) = 1$. A distribution on $\mathcal{X}$ is a weight-assignment D such that $D(x) \geq 0$ for all $x \in \mathcal{X}$. Consider the set of functions $H : \mathcal{X} \to \mathcal{Y}$ for sets $\mathcal{X}$ and $\mathcal{Y}$, denoted by $H_{\mathcal{X},\mathcal{Y}}$. We define the marginal weight assignment $D_{\mathcal{W}}$ of D on $H_{\mathcal{X},\mathcal{Y}}$ where the weight of a function $H_{\mathcal{W}} : \mathcal{W} \to \mathcal{Y}$ is equal to the sum of the weights of all $H \in H_{\mathcal{X},\mathcal{Y}}$ that agree with $H_{\mathcal{W}}$ on $\mathcal{W}$.

Definition 7 (k-wise equivalence). *We call two weight assignments D_1 and D_2 on $H_{\mathcal{X},\mathcal{Y}}$ k-wise equivalent if for all $\mathcal{W} \subseteq \mathcal{X}$ of size k, the marginal weight assignments $D_{1,\mathcal{W}}$ and $D_{2,\mathcal{W}}$ (of D_1 and D_2) over $H_{\mathcal{X},\mathcal{Y}}$ are identical.*

Definition 8 (k-wise independent function). *We call a function f k-wise independent function if f is k-wise equivalent to a random function.*

Lemma 1 (Theorem 3.1 in [36]). *Let A be a quantum algorithm making q quantum queries to an oracle $H : \mathcal{X} \to \mathcal{Y}$. If we draw H from some weight assignment D, then for every z, the quantity $\Pr_{H \leftarrow D}[A^H() = z]$ is a linear combination of the quantities $\Pr_{H \leftarrow D}[H(x_i) = r_i \forall i \in 1, \ldots, 2q]$ for all possible settings of the x_i and r_i.*

Lemma 2 (Theorem 6.1 in [36]). *If there exists $2q_i$-wise independent function, then any quantum algorithm A making q_i quantum queries to random oracles O_i can be efficiently simulated by a quantum algorithm B, which has the same output distribution, but makes no queries.*

Hence, a quantum algorithm B can simulate quantum random oracles in a polynomial-time. We use this simulation technique to simulate outputs of the hash function in the security proof of the SIDH UM protocol.

On the other hand, the other problem on security proofs in the quantum random oracle model is how to insert intended random values as the outputs of corresponding oracle inputs. Zhandry [36] showed a solution with the notion of semi-constant distributions $\mathbf{SC}_\omega$.

Definition 9 (Semi-constant distribution). *Define $\mathbf{SC}_\omega$, the semi-constant distribution, as the distribution over $H_{\mathcal{X},\mathcal{Y}}$ resulting from the following process:*

- *First, pick a random element y from $\mathcal{Y}$.*
- *For each $x \in \mathcal{X}$, do one of the following:*
  - *With probability ω, set $H(x) = y$. We call x a distinguished input to H.*
  - *Otherwise, set $H(x)$ to be a random element in $\mathcal{Y}$.*

Lemma 3 (Corollary 4.3 in [36]). *The distribution of outputs of a quantum algorithm making h queries to an oracle drawn from $\mathbf{SC}_\omega$ is at most a distance $\frac{3}{8}h^4\omega^2$ away from the case when the oracle is drawn from the uniform distribution.*

We suppose that the simulation succeeds with probability ϵ if the adversary uses an inserted random value as the outputs of corresponding oracle inputs. If the probability that the adversary uses one of the points is ω, then the simulation succeeds with probability $\epsilon\omega - \frac{3}{8}h^4\omega^2$. By choosing ω to maximize the success probability, the simulation succeeds with probability $O(\epsilon^2/h^4)$. We use this simulation technique to insert a SI-DDH instance into the hash function in the security proof of the SIDH UM protocol.

5.2 Description of SIDH UM Protocol

We give our SIDH UM protocol using the notation in Sect. 3.2. Public parameters are $\mathfrak{g} = (E;\ P_1,\ Q_1,\ P_2,\ Q_2)$ and $\mathfrak{e} = (\ell_1, \ell_2, e_1, e_2)$. We set Π = SIDHUM, that is, the protocol ID is "SIDHUM." Static and ephemeral keys are the same as our

$A_1 = \mathfrak{g}^{\mathfrak{a}_1}$		$B_1 = \mathfrak{g}^{\mathfrak{b}_1}$
$A_2 = \mathfrak{g}^{\mathfrak{a}_2}$		$B_2 = \mathfrak{g}^{\mathfrak{b}_2}$
$X = \mathfrak{g}^{\mathfrak{x}}$	$\xrightarrow{X}$	$Y = \mathfrak{g}^{\mathfrak{y}}$
	$\xleftarrow{Y}$	
$Z_1 = B_2{}^{\mathfrak{a}_1}$		$Z_1 = A_1{}^{\mathfrak{b}_2}$
$Z_2 = Y^{\mathfrak{x}}$		$Z_2 = X^{\mathfrak{y}}$
$K = H(\Pi, Z_1, Z_2, \hat{A}, \hat{B}, X, Y)$		

Fig. 5. Outline of SIDH UM Protocol.

$A_1 = \mathfrak{g}^{\mathfrak{a}_1}$		$B_1 = \mathfrak{g}^{\mathfrak{b}_1}$
$A_2 = \mathfrak{g}^{\mathfrak{a}_2}$		$B_2 = \mathfrak{g}^{\mathfrak{b}_2}$
$X = \mathfrak{g}^{\mathfrak{x}}$	$\xrightarrow{X}$	$Y = \mathfrak{g}^{\mathfrak{y}}$
	$\xleftarrow{Y}$	
$Z_1 = Y^{\mathfrak{a}_1}$		$Z_1 = A_1{}^{\mathfrak{y}}$
$Z_2 = B_2{}^{\mathfrak{x}}$		$Z_2 = X^{\mathfrak{b}_2}$
$Z_3 = B_2{}^{\mathfrak{a}_1}$		$Z_3 = A_1{}^{\mathfrak{b}_2}$
$Z_4 = Y^{\mathfrak{x}}$		$Z_4 = X^{\mathfrak{y}}$
$K = H(\Pi, Z_1, Z_2, Z_3, Z_4, \hat{A}, \hat{B}, X, Y)$		

Fig. 6. Outline of Biclique SIDH Protocol.

biclique SIDH protocol. Let two secret-key spaces for initiators and responders be given as $SK_1 = \mathbb{Z}/\ell_1^{e_1}\mathbb{Z}$ and $SK_2 = \mathbb{Z}/\ell_2^{e_2}\mathbb{Z}$, respectively.

User $\hat{A}$ has two static public keys, $A_1 = \mathfrak{g}^{\mathfrak{a}_1}$ and $A_2 = \mathfrak{g}^{\mathfrak{a}_2}$, where $\mathfrak{a}_1 = k_{\mathtt{A},1} \in_R SK_1$, $\mathfrak{a}_2 = k_{\mathtt{A},2} \in_R SK_2$, and $\mathfrak{a}_1$ and $\mathfrak{a}_2$ are $\hat{A}$'s static secret keys. User $\hat{B}$, also, has two static public keys, $B_1 = \mathfrak{g}^{\mathfrak{b}_1}$ and $B_2 = \mathfrak{g}^{\mathfrak{b}_2}$, where $\mathfrak{b}_1 = k_{\mathtt{B},1} \in_R SK_1$, $\mathfrak{b}_2 = k_{\mathtt{B},2} \in_R SK_2$, and $\mathfrak{b}_1$ and $\mathfrak{b}_2$ are $\hat{B}$'s static secret keys. Here, ephemeral secret keys for $\hat{A}$ and $\hat{B}$ are given as

$$\mathfrak{x} = k_{\mathtt{X}} \in_R SK_1, \text{ and } \mathfrak{y} = k_{\mathtt{Y}} \in_R SK_2,$$

respectively. $\hat{A}$ sends a ephemeral public key X as $X = g^{\mathfrak{x}}$ to $\hat{B}$, $\hat{B}$ sends back a ephemeral public key Y as $Y = g^{\mathfrak{y}}$ to $\hat{A}$.

$\hat{A}$ computes $Z_1 = B_2^{\mathfrak{a}_1}$, and $Z_2 = Y^{\mathfrak{x}}$, and then, obtains the session key K as $K = H(\Pi, Z_1, Z_2, \hat{A}, \hat{B}, X, Y)$, where H is a hash function.

$\hat{B}$ can computes the session key K as $K = H(\Pi, Z_1, Z_2, \hat{A}, \hat{B}, X, Y)$ from $Z_1 = A_1^{\mathfrak{b}_2}$, and $Z_2 = X^{\mathfrak{y}}$.

It is clear that the session keys of both parties are equal (Fig. 5).

5.3 Security

Theorem 1. *Suppose that H is modeled as a quantum random oracle and that the SI-DDH assumption hold for $(\mathfrak{g}, \mathfrak{e})$. Then the SIDH UM protocol is a post-quantum CK-secure authenticated key exchange protocol in the quantum random oracle model.*

In particular, for any AKE quantum adversary $\mathcal{M}$ against the SIDH UM protocol that runs in time at most t, involves at most n honest parties and activates at most s sessions, and makes at most h queries to the quantum random oracle and q SessionKeyReveal *queries, there exists an SI-DDH quantum adversary $\mathcal{S}$ such that*

$$\mathbf{Adv}^{\text{SI-DDH}}_{\mathfrak{g},\mathfrak{e}}(\mathcal{S}) \geq \frac{2\mathbf{Adv}^{\text{AKE}}_{\text{SIDHUM}}(\mathcal{M})^2}{n^2 s^2 (8hq + 3(h+q+1)^4)},$$

where $\mathcal{S}$ runs in time t plus time to perform $\mathcal{O}\big((n+s)\lambda\big)$ low-degree isogeny operations.

An intuition of the security proof is given in Sect. 5.1. The SI-DDH assumption used in Theorem 1 can be degree-sensitive. Hence, it implies security under

the SI-CDH assumption by using the reduction in Proposition 1. However, an additional reduction cost is necessary. It is not trivial to directly prove security under the SI-CDH assumption because of the no-cloning theorem. Specifically, in the reduction to the CK security, the SI-CDH solver wants to extract the answer of the SI-CDH problem from a random oracle query by the AKE adversary. However, the query is a quantum state, and the solver cannot record a copy of the input. Thus, this proof strategy does not work. Recently, Zhandry [37] introduced a technique to record quantum queries. How to apply this technique to the proof is an open problem.

6 Proposed Biclique SIDH Protocol

In this section, we propose the biclique SIDH protocol, where it can be proved in the random oracle model under the di-SI-GDH assumption.

It is worth to note here that the SIDH UM protocol is secure in the quantum random oracle model under the SI-DDH assumption, and therefore, the SIDH UM protocol is superior than the biclique SIDH protocol in the following points: the computational model of adversaries and the assumption relaying to the security. However, the biclique SIDH protocol can be shown to be secure in the CK^+ model, that is, the protocol resists against maximum exposure where a non-trivial combination of secret keys is revealed. This shows that the biclique SIDH protocol is superior than the SIDH UM protocol in this sense.

As our SIDH UM protocol in Sect. 5, the public parameter, $\mathfrak{g}$, contains two parameters, $(P_1,\ Q_1)$ and (P_2, Q_2) in our biclique SIDH protocol. A party has a key on $(P_1,\ Q_1)$ and the other key on (P_2, Q_2).

6.1 Description of Biclique SIDH Protocol

We give our biclique SIDH protocol using the notation in Sect. 3.2. Public parameters are $\mathfrak{g} = (E;\ P_1,\ Q_1,\ P_2,\ Q_2)$ and $\mathfrak{e} = (\ell_1, \ell_2, e_1, e_2)$. We set $\mathit{\Pi}$ = BCSIDH, that is, the protocol ID is "BCSIDH." Let two secret-key spaces for initiators and responders be given as $SK_1 = \mathbb{Z}/\ell_1^{e_1}\mathbb{Z}$ and $SK_2 = \mathbb{Z}/\ell_2^{e_2}\mathbb{Z}$, respectively.

User $\hat{A}$ has two static public keys, $A_1 = \mathfrak{g}^{\mathfrak{a}_1}$ and $A_2 = \mathfrak{g}^{\mathfrak{a}_2}$, where $\mathfrak{a}_1 = k_{\mathtt{A},1} \in_R SK_1$, $\mathfrak{a}_2 = k_{\mathtt{A},2} \in_R SK_2$, and $\mathfrak{a}_1$ and $\mathfrak{a}_2$ are $\hat{A}$'s static secret keys. User $\hat{B}$, also, has two static public keys, $B_1 = \mathfrak{g}^{\mathfrak{b}_1}$ and $B_2 = \mathfrak{g}^{\mathfrak{b}_2}$, where $\mathfrak{b}_1 = k_{\mathtt{B},1} \in_R SK_1$, $\mathfrak{b}_2 = k_{\mathtt{B},2} \in_R SK_2$, and $\mathfrak{b}_1$ and $\mathfrak{b}_2$ are $\hat{B}$'s static secret keys. Here, ephemeral secret keys for $\hat{A}$ and $\hat{B}$ are given as

$$\mathfrak{x} = k_{\mathtt{X}} \in_R SK_1, \text{ and } \mathfrak{y} = k_{\mathtt{Y}} \in_R SK_2,$$

respectively. $\hat{A}$ sends an ephemeral public key X as $X = g^{\mathfrak{x}}$ to $\hat{B}$, $\hat{B}$ sends back an ephemeral public key Y as $Y = g^{\mathfrak{y}}$ to $\hat{A}$.

$\hat{A}$ computes the non-trivial combinations of the ephemeral and static public keys as $Z_1 = Y^{\mathfrak{a}_1}$, $Z_2 = B_2^{\mathfrak{x}}$, $Z_3 = B_2^{\mathfrak{a}_1}$, and $Z_4 = Y^{\mathfrak{x}}$, and then, obtains

the session key K as $K = H(\Pi, Z_1, Z_2, Z_3, Z_4, \hat{A}, \hat{B}, X, Y)$, where H is a hash function.

$\hat{B}$ can computes the session key K as $K = H(\Pi, Z_1, Z_2, Z_3, Z_4, \hat{A}, \hat{B}, X, Y)$ from $Z_1 = A_1^{\mathfrak{y}}$, $Z_2 = X^{\mathfrak{b}_2}$, $Z_3 = A_1^{\mathfrak{b}_2}$, and $Z_4 = X^{\mathfrak{y}}$.

It is clear that the session keys of both parties are equal (Fig. 6).

Charles et al. [6] proposed a hash function secure against quantum adversaries from the isogeny computation intractability. Hence, we can use the isogeny-based hash function in the real implementation for H, however, H is modeled as a random oracle in the security proof below.

6.2 Security

Theorem 2. *Suppose that H is modeled as a random oracle and that the di-SI-GDH assumption hold for $(\mathfrak{g}, \mathfrak{e})$. Then the biclique SIDH protocol is a post-quantum CK$^+$-secure authenticated key exchange protocol in the random oracle model.*

In particular, for any AKE quantum adversary $\mathcal{M}$ against the biclique SIDH protocol that runs in time at most t, involves at most n honest parties and activate at most s sessions, and makes at most h queries to the random oracle, there exists a di-SI-GDH quantum adversary $\mathcal{S}$ such that

$$\mathbf{Adv}^{\text{di-SI-GDH}}_{\mathfrak{g},\mathfrak{e}}(\mathcal{S}) \geq \min\left\{\frac{1}{sn}, \frac{1}{n^2}, \frac{1}{s^2}\right\} \cdot \mathbf{Adv}^{\text{AKE}}_{\text{BCSIDH}}(\mathcal{M}),$$

where $\mathcal{S}$ runs in time t plus time to perform $\mathcal{O}\big((n+s)\lambda\big)$ low-degree isogeny operations and make $\mathcal{O}(h+s)$ queries to di-SI-DDH oracle.

As we consider a case where the security model is CK$^+$, an adversary may access to a non-trivial combination of secret keys. However, it means that the adversary cannot access to the other combination of the secret key. Thus, the di-SI-GDH solver can embedded an instance to the public keys where secret key are not revealed. As we assume the random oracle model, the adversary has to make a query which contains the di-SI-GDH answer, and then, the theorem can be proved. Note here that the di-SI-DDH oracle is necessary to keep consistency between the answers by the di-SI-GDH solver on adversary's questions.

We consider how to extend our security proof in the random oracle model to that in the *quantum* random oracle model as in the SIDH UM protocol. For completing the simulation, we need to extend the di-SI-GDH assumption (Definition 6). Namely, in random oracle simulation, $\mathcal{S}$ first checks compatibility of input elements using di-SI-DDH oracle. Hence, in the quantum ROM situation, since inputs are given in quantum superposition form, we should extend the di-SI-DDH oracle to take as input the superpositions. If the di-SI-GDH quantum adversary allows the extended di-SI-DDH oracle access, then our security proof can be converted to quantum ROM secure one.

7 Conclusion

We proposed two authenticated key exchange protocols from supersingular isogenies: SIDH UM and biclique SIDH. We also discussed a new approach for invalidating the Galbraith–Vercauteren attack for the gap problem on the supersingular isogeny Diffie–Hellman, and defined the di-SI-GDH assumption.

The SIDH UM protocol is secure in the CK and quantum random oracle models under the SI-DDH assumption. The biclique SIDH protocol is secure in the CK^+ and random oracle models under the di-SI-GDH assumption.

Our protocols are the first post-quantum one-round Diffie–Hellman type authenticated key exchange ones in the following points: one is secure under the quantum random oracle model and the other resists against maximum exposure where a non-trivial combination of secret keys is revealed.

References

1. Ambainis, A., Rosmanis, A., Unruh, D.: Quantum attacks on classical proof systems: The hardness of quantum rewinding. In: FOCS 2014. pp. 474–483 (2014)
2. Azarderakhsh, R., Jao, D., Kalach, K., Koziel, B., Leonardi, C.: Key compression for isogeny-based cryptosystems. In: AsiaPKC 2016. pp. 1–10 (2016)
3. Boneh, D., Dagdelen, Ö., Fischlin, M., Lehmann, A., Schaffner, C., Zhandry, M.: Random oracles in a quantum world. In: ASIACRYPT 2011. pp. 41–69 (2011)
4. Bos, J.W., Friedberger, S.: Fast arithmetic modulo 2^{x} $\mathrm{p}^{\mathrm{y}} \pm 1$. In: ARITH 2017. pp. 148–155 (2017)
5. Canetti, R., Krawczyk, H.: Analysis of key-exchange protocols and their use for building secure channels. In: EUROCRYPT 2001. pp. 453–474 (2001)
6. Charles, D., Lauter, K., Goren, E.: Cryptographic hash functions from expander graphs. J. Crypt. **22**(1), 93–113 (2009)
7. Childs, A., Jao, D., Soukharev, V.: Constructing elliptic curve isogenies in quantum subexponential time. J. Math. Crypt. **8**(1), 1–29 (2014)
8. Costello, C., Jao, D., Longa, P., Naehrig, M., Renes, J., Urbanik, D.: Efficient compression of SIDH public keys. In: EUROCRYPT 2017, I. pp. 679–706 (2017)
9. Costello, C., Longa, P., Naehrig, M.: Efficient algorithms for supersingular isogeny Diffie-Hellman. In: CRYPTO 2016, Part I. pp. 572–601 (2016)
10. Dagdelen, Ö., Fischlin, M., Gagliardoni, T.: The Fiat-Shamir transformation in a quantum world. In: ASIACRYPT 2013, Part II. pp. 62–81 (2013)
11. De Feo, L., Jao, D., Plût, J.: Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. J. Math. Crypt. **8**(3), 209–247 (2014)
12. Fujioka, A., Suzuki, K., Xagawa, K., Yoneyama, K.: Practical and post-quantum authenticated key exchange from one-way secure key encapsulation mechanism. In: ASIACCS 2013. pp. 83–94 (2013)
13. Fujioka, A., Suzuki, K., Xagawa, K., Yoneyama, K.: Strongly secure authenticated key exchange from factoring, codes, and lattices. Des. Codes Cryptography **76**(3), 469–504 (2015), a preliminary version appeared in PKC 2012 (2012)
14. Galbraith, S.: Mathematics of Public Key Cryptography. Cambridge Univ. Press (2012)
15. Galbraith, S.D.: Authenticated key exchange for SIDH. IACR Cryptology ePrint Archive **2018**, 266 (2018), `http://eprint.iacr.org/2018/266`

16. Galbraith, S.D., Petit, C., Shani, B., Ti, Y.B.: On the security of supersingular isogeny cryptosystems. In: ASIACRYPT 2016, Part I. pp. 63–91 (2016)
17. Galbraith, S.D., Vercauteren, F.: Computational problems in supersingular elliptic curve isogenies. IACR Cryptology ePrint Archive **2017**, 774 (2017), http://eprint.iacr.org/2017/774
18. Jao, D., Azarderakhsh, R., Campagna, M., Costello, C., Feo, L.D., Hess, B., Jalali, A., Koziel, B., LaMacchia, B., Longa, P., Naehrig, M., Renes, J., Soukharev, V., Urbanik, D.: Supersingular Isogeny Key Encapsulation (SIKE). submission to NIST Post-Quantum Cryptography Standardization (2017)
19. Jeong, I., Katz, J., Lee, D.: One-round protocols for two-party authenticated key exchange. In: ACNS 2004. pp. 220–232 (2004)
20. Koziel, B., Azarderakhsh, R., Kermani, M.M., Jao, D.: Post-quantum cryptography on FPGA based on isogenies on elliptic curves. IEEE Trans. on Circuits and Systems **64-I**(1), 86–99 (2017)
21. Koziel, B., Jalali, A., Azarderakhsh, R., Jao, D., Kermani, M.M.: NEON-SIDH: efficient implementation of supersingular isogeny Diffie-Hellman key exchange protocol on ARM. In: CANS 2016. pp. 88–103 (2016)
22. Krawczyk, H.: HMQV: A high-performance secure Diffie-Hellman protocol. In: CRYPTO 2005. pp. 546–566 (2005)
23. LeGrow, J., Jao, D., Azarderakhsh, R.: Modeling quantum-safe authenticated key establishment, and an isogeny-based protocol. IACR Cryptology ePrint Archive **2018**, 282 (2018), http://eprint.iacr.org/2018/282
24. Longa, P.: A note on post-quantum authenticated key exchange from supersingular isogenies. IACR Cryptology ePrint Archive **2018**, 267 (2018), http://eprint.iacr.org/2018/267
25. National Institute of Standards and Technology: Post-Quantum crypto standardization: Call for Proposals Announcement (December 2016), http://csrc.nist.gov/groups/ST/post-quantum-crypto/cfp-announce-dec2016.html
26. Petit, C.: Faster algorithms for isogeny problems using torsion point images. In: ASIACRYPT 2017, Part II. pp. 330–353 (2017)
27. Rostovtsev, A., Stolbunov, A.: Public-key cryptosystem based on isogenies. IACR Cryptology ePrint Archive **2006**, 145 (2006), http://eprint.iacr.org/2006/145
28. Shor, P.W.: Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM J. Comput. **26**(5), 1484–1509 (1997)
29. Silverman, J.: The Arithmetic of Elliptic Curves, GTM, vol. 106. Springer Verlag, 2nd edn. (2009)
30. Sutherland, A.: Identifying supersingular elliptic curves. LMS J. Comp. and Math. **15**, 317–325 (2012)
31. Thormarker, E.: Post-Quantum Cryptography: Supersingular Isogeny Diffie–Hellman Key Exchange. Master's thesis, Stockholm University (2017)
32. Urbanik, D., Jao, D.: SoK: The problem landscape of SIDH. In: APKC 2018. pp. 53–60 (2018)
33. Vélu, J.: Isogénies entre courbes elliptiques. C.R. Acad. Sc. Paris, Séries A. **273**, 238–241 (1971)
34. Xu, X., Xue, H., Wang, K., Tian, S., Liang, B., Yu, W.: Strongly secure authenticated key exchange from supersingular isogeny. IACR Cryptology ePrint Archive **2018**, 760 (2018)
35. Zhandry, M.: How to construct quantum random functions. In: FOCS 2012. pp. 679–687 (2012)
36. Zhandry, M.: Secure identity-based encryption in the quantum random oracle model. In: CRYPTO 2012. pp. 758–775 (2012)

37. Zhandry, M.: How to record quantum queries, and applications to quantum indifferentiability. IACR Cryptology ePrint Archive **2018** (2018)

A Proof of Theorem 1

Since H is modeled as a quantum random oracle, adversary $\mathcal{M}$ has only three ways to distinguish a session key of the test session from a random string.

- Guessing attack: $\mathcal{M}$ correctly guesses the session key.
- Key replication attack: $\mathcal{M}$ creates a session that is not matching to the test session, but has the same session key as the test session.
- Forging attack: $\mathcal{M}$ computes Z_1 and Z_2 used in the test session identified with $(\Pi, \mathcal{I}, \hat{A}, \hat{B}, X, Y)$, and queries H with a superposition including $(\Pi, Z_1, Z_2, \hat{A}, \hat{B}, X, Y)$.

Since H is a quantum random oracle, the probability of guessing the output of H is $\mathcal{O}(1/2^\lambda)$. Since non-matching sessions have different communicating parties or ephemeral public keys, key replication is equivalent to finding H-collision; therefore the probability of succeeding key replication is $\mathcal{O}(s^2/2^\lambda)$.

Let M be the event that $\mathcal{M}$ wins the security experiment with SIDHUM, H be the event that $\mathcal{M}$ succeeds forging attack, and $\overline{\mathsf{H}}$ the complementary event of H. Thus we have $\Pr[\mathsf{M} \mid \overline{\mathsf{H}}] = \frac{1}{2}$, and therefore $\mathbf{Adv}^{\mathrm{AKE}}_{\mathrm{SIDHUM}}(\mathcal{M}) = \Pr[\mathsf{M}] - \frac{1}{2} \leq \Pr[\mathsf{M} \cap \mathsf{H}]$.

By the definition of freshness in the CK-model, there are two cases that $\mathcal{M}$ chooses a test session.

- E_1: $\mathcal{M}$ chooses a test session without a matching session.
- E_2: $\mathcal{M}$ chooses a test session with a matching session, and reveals the static secret keys of both the owner of the test session and the owner of its matching session.

In each case, we will show how to construct an SI-DDH solver $\mathcal{S}$. Solver $\mathcal{S}$ is given an SI-DDH instance $(\mathsf{pk}^{\mathsf{sidh}}, U = \mathfrak{g}^{\mathfrak{u}}, V = \mathfrak{g}^{\mathfrak{v}}, W)$. Let R be an event that $\mathcal{M}$ chooses a test session whose owner and peer are the same party, and let $\overline{\mathsf{R}}$ be its complement.

Whether a certain event took place or not is decided at the end of the experiment. In other words for each event analysis bellow it is assumed that the event conditions are satisfied upon the adversary termination.

Before analyzing the events, we note that the session state of a session in the SIDH UM protocol is equivalent to the ephemeral secret key in the session as no other information (except the static secret key) is necessary to compute the shared secrets and the session key.

$\mathsf{E}_1 \cap \overline{\mathsf{R}}$. $\mathcal{S}$ prepares n honest parties, selects two honest parties $\hat{A}$ and $\hat{B}$ to whom $\mathcal{S}$ assigns the static public keys $A_1 = U$ and $B_2 = V$, and random static public and secret key pairs for A_2 and B_1. The remaining $n-2$ parties are assigned random static public and secret key pairs. $\mathcal{S}$ selects $i \in_R \{1, \ldots, s\}$, and chooses i-th

session $\mathtt{sid}^*$ among sessions, activated by $\mathcal{M}$, owned by $\hat{A}$ and having intended peer $\hat{B}$.

When $\mathcal{M}$ activates sessions between honest peers, $\mathcal{S}$ follows the protocol description. Since $\mathcal{S}$ knows static secret keys of at least one peer, it can respond all queries faithfully. The only exception is the session owned by $\hat{A}$ with the intended peer $\hat{B}$ because $\mathcal{S}$ does not know static secret keys of $\hat{A}$ and $\hat{B}$. Then, $\mathcal{S}$ sets W as Z_1 in such sessions.

Also, $\mathcal{S}$ chooses random $\mathfrak{r}^*$ and $\zeta \in \{0,1\}^\lambda$ as the ephemeral secret key and the session key of $\mathtt{sid}^*$, respectively. ζ is inserted as the output of H in the test session $\mathtt{sid}^*$ (i.e., the session key).

$\mathcal{S}$ has difficulty in responding hash queries because he/she needs to return superpositions corresponding to random values for exponentially many positions (The domain of H is $\mathbf{PRS} \times \mathbb{F}_{p^2}^2 \times \mathbf{IDS}^2 \times \mathit{SSEC}_{p,1} \times \mathit{SSEC}_{p,2}$). We solve this problem by using Lemma 2. Specifically, since the number of queries to H made by $\mathcal{M}$ is h for direct queries, q for $\mathsf{SessionKeyReveal}$ queries, and 1 for the Test query, for a total of $h+q+1$ queries, a $(h+q+1)$-wise independent function is sufficient to simulate superposition of outputs. There is the other difficulty to correctly answer the SI-DDH problem because $\mathcal{M}$ uses ζ with exponentially small probability if the position of ζ is only the corresponding input. We can also solve this problem by using Lemma 3. Specifically, the simulator inserts ζ for inputs $(\mathrm{pid}, Z_1, Z_2, \mathrm{uid}, \mathrm{uid}', \mathrm{epk}, \mathrm{epk}') \in \mathcal{X} \subset \mathbf{PRS} \times \mathbb{F}_{p^2}^2 \times \mathbf{IDS}^2 \times \mathit{SSEC}_{p,1} \times \mathit{SSEC}_{p,2}$. The probability that a randomly chosen input is contained in $\mathcal{X}$ is ω. If $\mathcal{M}$ chooses $(\mathrm{pid} = \Pi, Z_1 = W, Z_2 = Y^{*\mathfrak{r}}, \mathrm{uid} = \hat{A}, \mathrm{uid}' = \hat{B}, \mathrm{epk}, \mathrm{epk}') \in \mathcal{X}$ as the test session, then $\mathcal{S}$ can use the distinguishing capacity of $\mathcal{M}$ to distinguish the SI-DDH challenge.

We use the game hopping technique in the security proof. Let $\mathbf{Adv}^{\mathrm{AKE}}_{\mathrm{SIDHUM}}(\mathcal{M}, \mathbf{Game}_i)$ be the advantage of $\mathcal{M}$ in $\mathbf{Game}_i$.

- Let $\mathbf{Game}_0$ be the standard attack game for the CK security. When $\mathcal{M}$ poses a superposition to quantum random oracle H, the superposition of output values corresponding to the input is returned to $\mathcal{M}$. Then, $\mathbf{Adv}^{\mathrm{AKE}}_{\mathrm{SIDHUM}}(\mathcal{M}, \mathbf{Game}_0) = \mathbf{Adv}^{\mathrm{AKE}}_{\mathrm{SIDHUM}}(\mathcal{M})$.
- $\mathbf{Game}_1$ is the same as $\mathbf{Game}_0$ except that the game halts if $\mathcal{M}$ poses $\mathsf{Test}(\mathtt{sid})$ for $\mathtt{sid} \neq \mathtt{sid}^*$. Since $\mathtt{sid}^*$ is chosen from ns sessions, it holds that $\mathbf{Adv}^{\mathrm{AKE}}_{\mathrm{SIDHUM}}(\mathcal{M}, \mathbf{Game}_1) \geq \frac{1}{ns}\mathbf{Adv}^{\mathrm{AKE}}_{\mathrm{SIDHUM}}(\mathcal{M}, \mathbf{Game}_0)$.
- Let $\omega \in (0,1)$ be chosen later, and $\mathcal{X}$ be a subset of $\mathbf{PRS} \times \mathbb{F}_{p^2}^2 \times \mathbf{IDS}^2 \times \mathit{SSEC}_{p,1} \times \mathit{SSEC}_{p,2}$ where $(\mathrm{pid}, Z_1, Z_2, \mathrm{uid}, \mathrm{uid}', \mathrm{epk}, \mathrm{epk}') \in \mathbf{PRS} \times \mathbb{F}_{p^2}^2 \times \mathbf{IDS}^2 \times \mathit{SSEC}_{p,1} \times \mathit{SSEC}_{p,2}$ is put in $\mathcal{X}$ with independent probability ω. $\mathbf{Game}_2$ is the same as $\mathbf{Game}_1$ except that the game halts if $(\Pi, W, Y^{*\mathfrak{r}}, \hat{A}, \hat{B}, X^*, Y^*) \notin \mathcal{X}$ for the test session $\mathtt{sid}^* = (\Pi, \mathcal{I}, \hat{A}, \hat{B}, X^*, Y^*)$, $\mathcal{M}$ poses $\mathsf{SessionKeyReveal}(\Pi, \mathcal{I}, \mathrm{uid}, \mathrm{uid}', X', Y')$ such that $(\Pi, Z_1, Z_2, \mathrm{uid}, \mathrm{uid}', X', Y') \in \mathcal{X}$, or $\mathcal{M}$ poses $H(\Pi, Z_1, Z_2, \mathrm{uid}, \mathrm{uid}', X', Y')$ such that $(\Pi, Z_1, Z_2, \mathrm{uid}, \mathrm{uid}', X', Y') \in \mathcal{X}$. We note that Y^* is decided by $\mathcal{M}$ because $\mathtt{sid}^*$ has no matching session, and $\mathcal{M}$ cannot poses $\mathsf{SessionKeyReveal}(\Pi, \mathcal{I}, \hat{A}, \hat{B}, X^*, Y^*)$ by the freshness condition. $\mathbf{Adv}^{\mathrm{AKE}}_{\mathrm{SIDHUM}}(\mathcal{M}, \mathbf{Game}_2) \geq$

$\omega(1-\omega hq)\cdot \mathbf{Adv}^{\mathrm{AKE}}_{\mathrm{SIDHUM}}(\mathcal{M}, \mathbf{Game}_1) \geq \omega \mathbf{Adv}^{\mathrm{AKE}}_{\mathrm{SIDHUM}}(\mathcal{M}, \mathbf{Game}_1) - \omega^2 hq$ holds.

– $\mathbf{Game}_3$ is the same as $\mathbf{Game}_2$ except that ζ is set as $H(\mathrm{pid}, Z_1, Z_2, \mathrm{uid}, \mathrm{uid}', \mathrm{epk}, \mathrm{epk}')$ for all $(\mathrm{pid}, Z_1, Z_2, \mathrm{uid}, \mathrm{uid}', \mathrm{epk}, \mathrm{epk}') \in \mathcal{X}$, and choose $H(\mathrm{pid}, Z_1, Z_2, \mathrm{uid}, \mathrm{uid}', \mathrm{epk}, \mathrm{epk}')$ randomly for all other inputs. Now, H is distributed according to $\mathbf{SC}_\omega$. By Lemma 3, the output distribution of $\mathcal{M}$ in $\mathbf{Game}_3$ is at most a distance $\frac{3}{8}(h+q+1)^4\omega^2$ from that in $\mathbf{Game}_2$. Hence, $\mathbf{Adv}^{\mathrm{AKE}}_{\mathrm{SIDHUM}}(\mathcal{M}, \mathbf{Game}_3) \geq \mathbf{Adv}^{\mathrm{AKE}}_{\mathrm{SIDHUM}}(\mathcal{M}, \mathbf{Game}_2) - \frac{3}{8}(h+q+1)^4\omega^2$ holds.

Finally, we estimate $\mathbf{Adv}^{\mathrm{AKE}}_{\mathrm{SIDHUM}}(\mathcal{M}, \mathbf{Game}_3)$ by $\mathbf{Adv}^{\text{SI-DDH}}_{\mathfrak{g},\mathfrak{e}}(\mathcal{S})$ with the reduction to SI-DDH problem. For simplicity, we assume that $\mathcal{S}$ has quantum access to two random oracles $H_1 : \mathbf{PRS} \times \mathbb{F}_{p^2}^2 \times \mathbf{IDS}^2 \times SSEC_{p,1} \times SSEC_{p,2} \to \{0,1\}^\lambda$ and $H_2 : \mathbf{PRS} \times \mathbb{F}_{p^2}^2 \times \mathbf{IDS}^2 \times SSEC_{p,1} \times SSEC_{p,2} \to \{0,1\}$ where H_2 outputs 1 with probability ω. Let $\mathcal{X}$ be the set of $(\mathrm{pid}, Z_1, Z_2, \mathrm{uid}, \mathrm{uid}', \mathrm{epk}, \mathrm{epk}')$ such that $H_2(\mathrm{pid}, Z_1, Z_2, \mathrm{uid}, \mathrm{uid}', \mathrm{epk}, \mathrm{epk}') = 1$. We can see that the above conditions are equivalent to $\mathbf{Game}_3$. By Lemma 2, $\mathcal{S}$ can perfectly simulate H_1 and H_2 by using a $(h+q+1)$-wise independent function without oracle accesses. $\mathcal{S}$ prepares $\mathrm{R}^{\mathrm{list}}$ with entries of the form $(\mathrm{pid}, \mathrm{uid}, \mathrm{uid}', \mathrm{epk}, \mathrm{epk}', \mathrm{SK}) \in \mathbf{PRS} \times \mathbf{IDS}^2 \times SSEC_{p,1} \times SSEC_{p,2} \times \{0,1\}^\lambda$ and $\mathrm{H}^{\mathrm{list}}$ with entries of the form $(\mathrm{pid}, Z_1, Z_2, \mathrm{uid}, \mathrm{uid}', \mathrm{epk}, \mathrm{epk}', \mathrm{SK}) \in \mathbf{PRS} \times \mathbb{F}_{p^2}^2 \times \mathbf{IDS}^2 \times SSEC_{p,1} \times SSEC_{p,2} \times \{0,1\}^\lambda$, where pid is a string which gives a protocol identifier, and uid is a string which gives a user identifier, and $\mathcal{S}$ maintains two lists for consistent responses to H and $\mathsf{SessionKeyReveal}$ queries. On input $(\mathsf{pk}^{\mathsf{sidh}}, U, V, W)$, $\mathcal{S}$ works as follows:

– Choose $\mathfrak{x}^* \in_R SK_1$ and $\zeta \in_R \{0,1\}^\lambda$, and set $A_1 = U$, $B_2 = V$, and $X^* = \mathfrak{g}^{\mathfrak{x}^*}$ in $\mathtt{sid}^*$. The remaining $n-2$ parties are assigned random static public and secret key pairs. Set $\mathtt{sid}^* = (\Pi, \mathcal{I}, \hat{A}, \hat{B}, X^*, *)$.
– $\mathsf{Send}(\Pi, \mathcal{I}, \hat{A}, \hat{B})$: Solver $\mathcal{S}$ selects uniformly random ephemeral secret key $\mathfrak{x}$, computes ephemeral public key X honestly, records $(\Pi, \mathcal{I}, \hat{A}, \hat{B}, X)$ in List $\mathrm{R}^{\mathrm{list}}$, and returns X.
– $\mathsf{Send}(\Pi, \mathcal{R}, \hat{A}, \hat{B}, X)$: $\mathcal{S}$ selects uniformly random ephemeral secret key $\mathfrak{y}$, computes ephemeral public key Y honestly, records $(\Pi, \mathcal{R}, \hat{A}, \hat{B}, X, Y)$ in List $\mathrm{R}^{\mathrm{list}}$ as completed, and returns Y.
– $\mathsf{Send}(\Pi, \mathcal{I}, \hat{A}, \hat{B}, X, Y)$: If session $(\Pi, \mathcal{I}, \hat{A}, \hat{B}, X)$ is not recorded in List $\mathrm{R}^{\mathrm{list}}$, $\mathcal{S}$ records session $(\Pi, \mathcal{I}, \hat{A}, \hat{B}, X, Y)$ in List $\mathrm{R}^{\mathrm{list}}$ as not completed. Otherwise, $\mathcal{S}$ records the session in List $\mathrm{R}^{\mathrm{list}}$ as completed.
– $H(\cdot)$: $\mathcal{S}$ simulates a random oracle such that

$$H(\Pi, Z_1, Z_2, \hat{A}, \hat{B}, X, Y) = \begin{cases} \zeta & \text{if } H_2(\Pi, Z_1, Z_2, \hat{A}, \hat{B}, X, Y) = 1 \\ H_1(\Pi, Z_1, Z_2, \hat{A}, \hat{B}, X, Y) & \text{otherwise} \end{cases}$$

– $\mathsf{SessionKeyReveal}(\cdot)$: When $\mathcal{M}$ poses $(\Pi, \mathcal{I}, \mathrm{uid}, \mathrm{uid}', X, Y)$ such that $H_2(\Pi, Z_1, Z_2, \mathrm{uid}, \mathrm{uid}', X, Y) = 1$, then outputs a random bit and aborts. Otherwise, return $\mathrm{SK} = H_1(\mathrm{pid}, Z_1, Z_2, \mathrm{uid}, \mathrm{uid}', X, Y)$.

– $\mathsf{SessionStateReveal}(\mathtt{sid})$: $\mathcal{S}$ responds to the query faithfully.
– $\mathsf{Corrupt}(\hat{C})$: If $\hat{C}$ is queried before, $\mathcal{S}$ returns error. Otherwise, $\mathcal{S}$ responds to the query faithfully. Note that $\mathsf{Corrupt}(\hat{A})$ nor $\mathsf{Corrupt}(\hat{B})$ is never posed by the freshness condition.
– $\mathsf{Test}(\mathtt{sid})$: If $\mathtt{sid} \neq \mathtt{sid}^*$, then $\mathcal{S}$ aborts with failure. Otherwise, $\mathcal{S}$ responds ζ to the query.
– If adversary $\mathcal{M}$ outputs guess γ, $\mathcal{S}$ outputs γ.

$\mathcal{S}$ may abort in the simulation of $\mathsf{SessionKeyReveal}$ and Test. Also, $\mathcal{S}$ may fail if $\mathcal{M}$ poses $H(\Pi, Z_1, Z_2, \mathrm{uid}, \mathrm{uid}', X, Y)$ such that $(\Pi, Z_1, Z_2, \mathrm{uid}, \mathrm{uid}', X, Y) \in \mathcal{X}$. However, in $\mathbf{Game}_3$, these events do not occur because of the game hopping. In the case of $W = (\mathfrak{g}^{\mathfrak{u}})^{\mathfrak{v}}$, the simulation of Test query is the same as the real session key. In the case of $W = (\mathfrak{g}^{\mathfrak{r}})^{\mathfrak{s}}$ with random secret keys $\mathfrak{r}$ and $\mathfrak{s}$, the simulation of Test query is the same as the random session key. Thus, $\mathbf{Adv}^{\mathrm{AKE}}_{\mathrm{SIDHUM}}(\mathcal{M}, \mathbf{Game}_3)$ is

$$\mathbf{Adv}^{\mathrm{AKE}}_{\mathrm{SIDHUM}}(\mathcal{M}, \mathbf{Game}_3) = \mathbf{Adv}^{\text{SI-DDH}}_{\mathfrak{g},\mathfrak{e}}(\mathcal{S}).$$

Therefore, $\mathbf{Adv}^{\mathrm{AKE}}_{\mathrm{SIDHUM}}(\mathcal{M})$ is

$$\mathbf{Adv}^{\mathrm{AKE}}_{\mathrm{SIDHUM}}(\mathcal{M}) \leq \frac{ns}{\omega}\mathbf{Adv}^{\text{SI-DDH}}_{\mathfrak{g},\mathfrak{e}}(\mathcal{S}) + ns\omega\left(hq + \frac{3}{8}(h+q+1)^4\right).$$

The right side is minimized when $\omega = \frac{4\mathbf{Adv}^{\mathrm{AKE}}_{\mathrm{SIDHUM}}(\mathcal{M})}{ns(8hq+3(h+q+1)^4)}$.

$\mathsf{E}_1 \cap \mathsf{R}$. $\mathcal{S}$ prepares n honest parties, selects an honest party $\hat{A}$ and assigns its static public keys as $A_1 = U$ and $A_2 = V$. The proof is almost the same as in $\mathsf{E}_1 \cap \overline{\mathsf{R}}$. The party $\hat{A}$ is simulated as $\hat{B}$ in $\mathsf{E}_1 \cap \overline{\mathsf{R}}$.

E_2. $\mathcal{S}$ prepares n honest parties, selects two honest parties $\hat{A}$ and $\hat{B}$, and assigns random static public and secret key pairs for all parties (i.e., $\mathcal{S}$ knows $\mathfrak{a}_1$ and $\mathfrak{b}_2$). $\mathcal{S}$ selects $i \in_R \{1, \ldots, s\}$, and chooses i-th session $\mathtt{sid}^*$ among sessions, activated by $\mathcal{M}$, owned by $\hat{A}$ and having intended peer $\hat{B}$.

When $\mathcal{M}$ activates sessions between honest peers, $\mathcal{S}$ follows the protocol description. Since $\mathcal{S}$ knows static secret keys of at least one peer, it can respond all queries faithfully. In $\mathtt{sid}^*$, $\mathcal{S}$ assigns ephemeral public keys $X^* = U$ and $Y^* = V$ of $\hat{A}$ and $\hat{B}$, respectively. Then, $\mathcal{S}$ sets W as Z_2 in $\mathtt{sid}^*$. Also, $\mathcal{S}$ chooses random $\zeta \in \{0,1\}^\lambda$ as the session key of $\mathtt{sid}^*$. ζ is inserted as the output of H in the test session $\mathtt{sid}^*$ (i.e., the session key).

We use the game hopping technique in the security proof. Let $\mathbf{Adv}^{\mathrm{AKE}}_{\mathrm{SIDHUM}}(\mathcal{M}, \mathbf{Game}_i)$ be the advantage of $\mathcal{M}$ in $\mathbf{Game}_i$.

– Let $\mathbf{Game}_0$ be the standard attack game for the CK security. When $\mathcal{M}$ poses a superposition to quantum random oracle H, the superposition of output values corresponding to the input is returned to $\mathcal{M}$. Then, $\mathbf{Adv}^{\mathrm{AKE}}_{\mathrm{SIDHUM}}(\mathcal{M}, \mathbf{Game}_0) = \mathbf{Adv}^{\mathrm{AKE}}_{\mathrm{SIDHUM}}(\mathcal{M})$.
– $\mathbf{Game}_1$ is the same as the game, $\mathbf{Game}_0$, except that the game halts if $\mathcal{M}$ poses $\mathsf{Test}(\mathtt{sid})$ for $\mathtt{sid} \neq \mathtt{sid}^*$. Since $\mathtt{sid}^*$ is chosen from ns sessions, $\mathbf{Adv}^{\mathrm{AKE}}_{\mathrm{SIDHUM}}(\mathcal{M}, \mathbf{Game}_1) \geq \frac{1}{ns}\mathbf{Adv}^{\mathrm{AKE}}_{\mathrm{SIDHUM}}(\mathcal{M}, \mathbf{Game}_0)$ holds.

– $\mathbf{Game}_2$ is the same as $\mathbf{Game}_1$ except that ζ is set as $H(\mathrm{pid}, Z_1^* = B_2^{\mathfrak{a}_1}, W, \hat{A}, \hat{B}, U, V)$, and choose $H(\mathrm{pid}, Z_1, Z_2, \hat{A}, \hat{B}, X, Y)$ randomly for all other inputs. Now, H is distributed according to $\mathbf{SC}_\omega$ where ω is the probability of randomly selecting $(\mathrm{pid}, Z_1^*, W, \hat{A}, \hat{B}, U, V)$ from the domain, which is negligibly small. By Lemma 3, the output distribution of $\mathcal{M}$ in $\mathbf{Game}_2$ is at most a distance $\frac{3}{8}(h+q+1)^4\omega^2$ from that in $\mathbf{Game}_1$. Hence, $\mathbf{Adv}^{\mathrm{AKE}}_{\mathrm{SIDHUM}}(\mathcal{M}, \mathbf{Game}_2) \geq \mathbf{Adv}^{\mathrm{AKE}}_{\mathrm{SIDHUM}}(\mathcal{M}, \mathbf{Game}_1) - \frac{3}{8}(h+q+1)^4\omega^2 = \mathbf{Adv}^{\mathrm{AKE}}_{\mathrm{SIDHUM}}(\mathcal{M}, \mathbf{Game}_1) - \mathit{negl}$ holds.

Finally, we estimate $\mathbf{Adv}^{\mathrm{AKE}}_{\mathrm{SIDHUM}}(\mathcal{M}, \mathbf{Game}_2)$ by using $\mathbf{Adv}^{\text{SI-DDH}}_{\mathfrak{g},\mathfrak{e}}(\mathcal{S})$. For simplicity, we assume that $\mathcal{S}$ has quantum access to two random oracles $H_1 : \mathbf{PRS} \times \mathbb{F}_{p^2}^2 \times \mathbf{IDS}^2 \times \mathit{SSEC}_{p,1} \times \mathit{SSEC}_{p,2} \to \{0,1\}^\lambda$ and $H_2 : \mathbf{PRS} \times \mathbb{F}_{p^2}^2 \times \mathbf{IDS}^2 \times \mathit{SSEC}_{p,1} \times \mathit{SSEC}_{p,2} \to \{0,1\}$ where H_2 outputs 1 with probability ω. By Lemma 2, $\mathcal{S}$ can perfectly simulate H_1 and H_2 by using a $(h+q+1)$-wise independent function without oracle accesses. $\mathcal{S}$ prepares $\mathrm{R}^{\mathrm{list}}$ with entries of the form $(\mathrm{pid}, \mathrm{uid}, \mathrm{uid}', \mathrm{epk}, \mathrm{epk}', \mathrm{SK}) \in \mathbf{PRS} \times \mathbf{IDS}^2 \times \mathit{SSEC}_{p,1} \times \mathit{SSEC}_{p,2} \times \{0,1\}^\lambda$ and $\mathrm{H}^{\mathrm{list}}$ with entries of the form $(\mathrm{pid}, Z_1, Z_2, \mathrm{uid}, \mathrm{uid}', \mathrm{epk}, \mathrm{epk}', \mathrm{SK}) \in \mathbf{PRS} \times \mathbb{F}_{p^2}^2 \times \mathbf{IDS}^2 \times \mathit{SSEC}_{p,1} \times \mathit{SSEC}_{p,2} \times \{0,1\}^\lambda$, where pid is a string which gives a protocol identifier, and uid is a string which gives an user identifier, and $\mathcal{S}$ maintains two lists for consistent responses to H and $\mathsf{SessionKeyReveal}$ queries. On input $(\mathsf{pk}^{\mathsf{sidh}}, U, V, W)$, $\mathcal{S}$ works as follows:

– Choose $\mathfrak{a}_1 \in_R SK_1$, $\mathfrak{b}_2 \in_R SK_2$ and $\zeta \in_R \{0,1\}^\lambda$, and set $A_1 = \mathfrak{g}^{\mathfrak{a}_1}$ and $B_2 = \mathfrak{g}^{\mathfrak{b}_2}$, $X^* = U$ and $Y^* = V$ in $\mathtt{sid}^*$. n parties are assigned random static public and secret key pairs. Set $\mathtt{sid}^* = (\Pi, \mathcal{I}, \hat{A}, \hat{B}, X^*, Y^*)$.
– $\mathsf{Send}(\Pi, \mathcal{I}, \hat{A}, \hat{B})$: Solver $\mathcal{S}$ selects uniformly random ephemeral secret key $\mathfrak{x}$, computes ephemeral public key X honestly, records $(\Pi, \mathcal{I}, \hat{A}, \hat{B}, X)$ in List $\mathrm{R}^{\mathrm{list}}$, and returns X.
– $\mathsf{Send}(\Pi, \mathcal{R}, \hat{A}, \hat{B}, X)$: $\mathcal{S}$ selects uniformly random ephemeral secret key $\mathfrak{y}$, computes ephemeral public key Y honestly, records $(\Pi, \mathcal{R}, \hat{A}, \hat{B}, X, Y)$ in List $\mathrm{R}^{\mathrm{list}}$ as completed, and returns Y.
– $\mathsf{Send}(\Pi, \mathcal{I}, \hat{A}, \hat{B}, X, Y)$: If session $(\Pi, \mathcal{I}, \hat{A}, \hat{B}, X)$ is not recorded in List $\mathrm{R}^{\mathrm{list}}$, $\mathcal{S}$ records session $(\Pi, \mathcal{I}, \hat{A}, \hat{B}, X, Y)$ in List $\mathrm{R}^{\mathrm{list}}$ as not completed. Otherwise, $\mathcal{S}$ records the session in List $\mathrm{R}^{\mathrm{list}}$ as completed.
– $H(\cdot)$: $\mathcal{S}$ simulates a random oracle such that

$$H(\Pi, Z_1, Z_2, \hat{A}, \hat{B}, X, Y) = \begin{cases} \zeta & \text{if } H_2(\Pi, Z_1, Z_2, \hat{A}, \hat{B}, X, Y) = 1 \\ H_1(\Pi, Z_1, Z_2, \hat{A}, \hat{B}, X, Y) & \text{otherwise} \end{cases}$$

– $\mathsf{SessionKeyReveal}(\cdot)$: When $\mathcal{M}$ poses $(\Pi, \mathcal{I}, \hat{A}, \hat{B}, X, Y)$ such that $H_2(\Pi, Z_1, Z_2, \hat{A}, \hat{B}, X, Y) = 1$, then outputs a random bit and aborts. Otherwise, return $\mathrm{SK} = H_1(\Pi, Z_1, Z_2, \hat{A}, \hat{B}, X, Y)$.
– $\mathsf{SessionStateReveal}(\mathtt{sid})$: $\mathcal{S}$ responds to the query faithfully.
– $\mathsf{Corrupt}(\hat{C})$: If $\hat{C}$ is queried before, $\mathcal{S}$ returns error. Otherwise, $\mathcal{S}$ responds to the query faithfully. Note that $\mathfrak{a}_1$ and $\mathfrak{b}_2$ are given to $\mathcal{M}$.

– $\mathsf{Test}(\mathtt{sid})$: If $\mathtt{sid} \neq \mathtt{sid}^*$, then $\mathcal{S}$ aborts with failure. Otherwise, $\mathcal{S}$ responds ζ to the query.
– If adversary $\mathcal{M}$ outputs guess γ, $\mathcal{S}$ outputs γ.

$\mathcal{S}$ may abort in the simulation of $\mathsf{SessionKeyReveal}$ and Test. However, in $\mathbf{Game}_2$, these events do not occur because of the game hopping. In the case of $W = (\mathfrak{g}^{\mathfrak{u}})^{\mathfrak{v}}$, the simulation of Test query is the same as the real session key. In the case of $W = (\mathfrak{g}^{\mathfrak{r}})^{\mathfrak{s}}$ with random secret keys $\mathfrak{r}$ and $\mathfrak{s}$, the simulation of Test query is the same as the random session key. Thus, $\mathbf{Adv}^{\mathrm{AKE}}_{\mathrm{SIDHUM}}(\mathcal{M}, \mathbf{Game}_2)$ is

$$\mathbf{Adv}^{\mathrm{AKE}}_{\mathrm{SIDHUM}}(\mathcal{M}, \mathbf{Game}_2) = \mathbf{Adv}^{\text{SI-DDH}}_{\mathfrak{g},\mathfrak{e}}(\mathcal{S}).$$

Therefore, $\mathbf{Adv}^{\mathrm{AKE}}_{\mathrm{SIDHUM}}(\mathcal{M})$ is

$$\mathbf{Adv}^{\mathrm{AKE}}_{\mathrm{SIDHUM}}(\mathcal{M}) \leq ns \cdot \mathbf{Adv}^{\text{SI-DDH}}_{\mathfrak{g},\mathfrak{e}}(\mathcal{S}) + negl.$$

□

B Proof of Theorem 2

Since H is modeled as a random oracle, adversary $\mathcal{M}$ has only three ways to distinguish a session key of the test session from a random string.

– Guessing attack: $\mathcal{M}$ correctly guesses the session key.
– Key replication attack: $\mathcal{M}$ creates a session that is not matching to the test session, but has the same session key as the test session.
– Forging attack: $\mathcal{M}$ computes Z_1, Z_2, Z_3, and Z_4 used in the test session identified with (Π, $\mathcal{I}$, $\hat{A}$, $\hat{B}$, X, Y), and queries H with (Π, Z_1, Z_2, Z_3, Z_4, $\hat{A}$, $\hat{B}$, X, Y).

Since H is a random oracle, the probability of guessing the output of H is $\mathcal{O}(1/2^{\lambda})$. Since non-matching sessions have different communicating parties or ephemeral public keys, key replication is equivalent to finding H-collision; therefore the probability of succeeding key replication is $\mathcal{O}(s^2/2^{\lambda})$. However to detect collision the adversary has to query with both inputs the random oracle, in particular query with Z_1, Z_2, Z_3, and Z_4 used in the test session as describe in Forging attack above.

Let M be the event that $\mathcal{M}$ wins the security experiment with BCSIDH, H be the event that $\mathcal{M}$ succeeds forging attack, and $\overline{\mathsf{H}}$ the complementary event of H. Thus we have $\Pr[\mathsf{M} \mid \overline{\mathsf{H}}] = \frac{1}{2}$, and therefore

$$\mathbf{Adv}^{\mathrm{AKE}}_{\mathrm{BCSIDH}}(\mathcal{M}) = \Pr[\mathsf{M}] - \frac{1}{2} \leq \Pr[\mathsf{M} \cap \mathsf{H}]. \tag{1}$$

By the definition of freshness in the CK$^+$-model, there are six cases that $\mathcal{M}$ chooses a test session.

– E_1: $\mathcal{M}$ chooses a test session without a matching session, and does not reveal the ephemeral secret key of the owner of the test session.

– E_2: $\mathcal{M}$ chooses a test session without a matching session, and does not reveal the static secret key of the owner of the test session.
– E_3: $\mathcal{M}$ chooses a test session with a matching session, and does not reveal the ephemeral secret key of both the owner of the test session and the owner of its matching session.
– E_4: $\mathcal{M}$ chooses a test session with a matching session, and does not reveal the static secret keys of both the owner of the test session and the owner of its matching session.
– E_5: $\mathcal{M}$ chooses a test session with a matching session, and does not reveal the ephemeral secret key of the owner of the test session and the static secret key of the owner of its matching session.
– E_6: $\mathcal{M}$ chooses a test session with a matching session, and does not reveal the static secret key of the owner of the test session and the ephemeral secret key of the owner of its matching session.

In each case, we will show how to construct a di-SI-GDH solver $\mathcal{S}$. Solver $\mathcal{S}$ is given an SI-CDH instance $(\mathsf{pk}^{\mathsf{sidh}}, U, V)$. Let R be an event that $\mathcal{M}$ chooses a test session whose owner and peer are the same party, and let $\overline{\mathsf{R}}$ be its complement.

Whether a certain event took place or not is decided at the end of the experiment. In other words for each event analysis bellow it is assumed that the event conditions are satisfied upon the adversary termination.

Before analyzing the events, we note that the session state of a session in the biclique SIDH protocol is equivalent to the ephemeral secret key in the session as no other information (except the static secret key) is necessary to compute the shared secrets and the session key.

$\mathsf{E}_1 \cap \overline{\mathsf{R}}$. $\mathcal{S}$ prepares n honest parties, selects one party $\hat{B}$ to whom $\mathcal{S}$ assigns the static public key $B_2 = V$ and a random static public and secret key pair for B_1. The remaining $n-1$ parties are assigned random static public and secret key pairs. $\mathcal{S}$ selects $i \in_R \{1, \ldots, s\}$, and chooses i-th session $\mathtt{sid}^*$ among sessions, activated by $\mathcal{M}$ and owned by an honest party different from $\hat{B}$.

When $\mathcal{M}$ activates sessions between honest peers, $\mathcal{S}$ follows the protocol description. Since $\mathcal{S}$ knows static secret keys of at least one peer, it can respond all queries faithfully. The only exception is the session $\mathtt{sid}^*$, for which $\mathcal{S}$ sets ephemeral public key of $\mathtt{sid}^*$ to U, and chooses a random $\zeta \in \{0,1\}^\lambda$ as the session key of $\mathtt{sid}^*$.

The simulator has difficulty in responding queries related to $\hat{B}$ because $\mathcal{S}$ does not know one of the static secret keys of $\hat{B}$. More precisely, for sessions owned by $\hat{B}$ with a peer $\hat{C}$ controlled by $\mathcal{M}$, $\mathcal{S}$ cannot compute the shared secrets, Z_1, Z_2, Z_3, and Z_4, but may have to answer $\mathsf{SessionKeyReveal}$ queries. $\mathcal{M}$ could also derive session keys of these session by computing the shared secrets, Z_1, Z_2, Z_3, and Z_4, and query H. If four values do not coincide, then $\mathcal{S}$ fails its simulation. To handle this situations, $\mathcal{S}$ prepares $\mathrm{R}^{\mathrm{list}}$ with entries of the form (pid, rid, uid, uid$'$, W, W', SK) $\in \mathbf{PRS} \times \{0,1\} \times \mathbf{IDS}^2 \times \mathit{SSEC}_{p,1} \times \mathit{SSEC}_{p,2} \times \{0,1\}^\lambda$ and $\mathrm{H}^{\mathrm{list}}$ with entries of the form (pid, Z_1, Z_2, Z_3, Z_4, uid, uid$'$, W, W', SK) $\in \mathbf{PRS} \times \mathbb{F}_{p^2}^4 \times \mathbf{IDS}^2 \times \mathit{SSEC}_{p,1} \times \mathit{SSEC}_{p,2} \times \{0,1\}^\lambda$, where pid is a string which gives a protocol identifier, rid is a bit which gives a role identifier, e.g., 0 in $\mathcal{I}$

and 1 in $\mathcal{R}$, and uid is a string which gives a user identifier, and $\mathcal{S}$ maintains two lists for consistent responses to H and SessionKeyReveal queries as follows. Below, Y is generated by $\mathcal{S}$ on behalf of $\hat{B}$.

- $\mathsf{Send}(\Pi, \mathcal{I}, \hat{A}, \hat{B})$: Solver $\mathcal{S}$ selects uniformly random ephemeral secret key $\mathfrak{x}$, computes ephemeral public key X honestly, records $(\Pi, \mathcal{I}, \hat{A}, \hat{B}, X)$ in List $\mathrm{R}^{\mathrm{list}}$, and returns X.
- $\mathsf{Send}(\Pi, \mathcal{R}, \hat{A}, \hat{B}, X)$: $\mathcal{S}$ selects uniformly random ephemeral secret key $\mathfrak{y}$, computes ephemeral public key Y honestly, records $(\Pi, \mathcal{R}, \hat{A}, \hat{B}, X, Y)$ in List $\mathrm{R}^{\mathrm{list}}$ as completed, and returns Y.
- $\mathsf{Send}(\Pi, \mathcal{I}, \hat{A}, \hat{B}, X, Y)$: If session $(\Pi, \mathcal{I}, \hat{A}, \hat{B}, X)$ is not recorded in List $\mathrm{R}^{\mathrm{list}}$, $\mathcal{S}$ records session $(\Pi, \mathcal{I}, \hat{A}, \hat{B}, X, Y)$ in List $\mathrm{R}^{\mathrm{list}}$ as not completed. Otherwise, $\mathcal{S}$ records the session in List $\mathrm{R}^{\mathrm{list}}$ as completed.
- $H(\cdot)$: $\mathcal{S}$ simulates a random oracle in the usual way except for queries of the form $(\Pi, Z_1, Z_2, Z_3, Z_4, \hat{B}, \hat{C}, Y, X)$ and $(\Pi, Z_1, Z_2, Z_3, Z_4, \hat{C}, \hat{B}, X, Y)$. When $(\Pi, Z_1, Z_2, Z_3, Z_4, \hat{B}, \hat{C}, Y, X)$ is queried, $\mathcal{S}$ responds to these queries in the following way: (when $(\Pi, Z_1, Z_2, Z_3, Z_4, \hat{C}, \hat{B}, X, Y)$ is queried, $\mathcal{S}$ responds in a similar way)
  - if $(\Pi, Z_1, Z_2, Z_3, Z_4, \hat{B}, \hat{C}, Y, X, \mathrm{SK}) \in \mathrm{H}^{\mathrm{list}}$ for some SK, $\mathcal{S}$ returns SK to $\mathcal{M}$.
  - else if
    – the validity conditions, $\text{SI-DDH}(B_1, X, Z_1) = 1$, $\text{SI-DDH}(Y, C_2, Z_2) = 1$, $\text{SI-DDH}(B_1, C_2, Z_3) = 1$, and $\text{SI-DDH}(Y, X, Z_4) = 1$, hold,
    then if there exists $(\Pi, \mathcal{I}, \hat{B}, \hat{C}, Y, X, \mathrm{SK}) \in \mathrm{R}^{\mathrm{list}}$, $\mathcal{S}$ returns SK; otherwise, $\mathcal{S}$ chooses $\mathrm{SK} \in_R \{0,1\}^\lambda$, returns SK and stores $(\Pi, \mathcal{I}, \hat{B}, \hat{C}, Y, X, \mathrm{SK})$ in $\mathrm{R}^{\mathrm{list}}$. $\mathcal{S}$ also stores the new tuple $(\Pi, Z_1, Z_2, Z_3, Z_4, \hat{B}, \hat{C}, Y, X, \mathrm{SK})$ in $\mathrm{H}^{\mathrm{list}}$.
  - else $\mathcal{S}$ choose $\mathrm{SK} \in_R \{0,1\}^\lambda$, returns it to $\mathcal{M}$ and stores the new tuple $(\Pi, Z_1, Z_2, Z_3, Z_4, \hat{B}, \hat{C}, Y, X, \mathrm{SK})$ in $\mathrm{H}^{\mathrm{list}}$.
- $\mathsf{SessionKeyReveal}(\cdot)$: $\mathcal{S}$ simulates these queries in the usual way except for queries of the form $(\Pi, \mathcal{I}, \hat{B}, \hat{C}, Y, X)$ and $(\Pi, \mathcal{R}, \hat{B}, \hat{C}, X, Y)$. When $(\Pi, \mathcal{I}, \hat{B}, \hat{C}, Y, X)$ is queried, $\mathcal{S}$ does one of the following: (when $(\Pi, \mathcal{R}, \hat{B}, \hat{C}, X, Y)$ is queried, $\mathcal{S}$ responds in a similar way)
  - if there is no session with identifier $(\Pi, \mathcal{I}, \hat{B}, \hat{C}, Y, X)$, the query is aborted.
  - else if $(\Pi, \mathcal{I}, \hat{B}, \hat{C}, Y, X, \mathrm{SK}) \in \mathrm{R}^{\mathrm{list}}$ for some SK, $\mathcal{S}$ returns SK to $\mathcal{M}$.
  - else if $(\Pi, Z_1, Z_2, Z_3, Z_4, \hat{B}, \hat{C}, Y, X, \mathrm{SK}) \in \mathrm{H}^{\mathrm{list}}$ such that $\text{SI-DDH}(B_1, X, Z_1) = 1$, $\text{SI-DDH}(Y, C_2, Z_2) = 1$, $\text{SI-DDH}(B_1, C_2, Z_3) = 1$, and $\text{SI-DDH}(Y, X, Z_4) = 1$, $\mathcal{S}$ returns SK and stores the new tuple $(\Pi, \mathcal{I}, \hat{B}, \hat{C}, Y, X, \mathrm{SK})$ in $\mathrm{R}^{\mathrm{list}}$.
- $\mathsf{SessionStateReveal}(\mathtt{sid})$: If the corresponding ephemeral public key is U, then solver $\mathcal{S}$ aborts with failure. Otherwise, solver $\mathcal{S}$ responds to the query faithfully.
- $\mathsf{Corrupt}(\hat{A})$: If $\hat{A}$ is queried before, solver $\mathcal{S}$ returns error. Otherwise, solver $\mathcal{S}$ responds to the query faithfully.
- $\mathsf{Test}(\mathtt{sid})$: If $\mathtt{sid}$ is not the i-th session, owned by $\hat{A}$, then solver $\mathcal{S}$ aborts with failure. Otherwise, solver $\mathcal{S}$ responds to the query faithfully.

– If adversary $\mathcal{M}$ outputs guess γ, solver $\mathcal{S}$ aborts with failure.

Provided that $\mathsf{E}_1 \cap \overline{\mathsf{R}}$ occurs and $\mathcal{M}$ selects $\mathtt{sid}^*$ as the test session with peer $\hat{B}$, the simulation does not fail. In this case, the session identifier of $\mathtt{sid}^*$ is $(\Pi, \mathcal{I}, \hat{A}, \hat{B}, U, Y)$, where Y is the incoming ephemeral public key of $\mathtt{sid}^*$. If $\mathcal{M}$ wins the security game, it must have queried H with inputs $Z_1 = \text{SI-CDH}(A_1, Y)$, $Z_2 = \text{SI-CDH}(U, B_2)$, $Z_3 = \text{SI-CDH}(A_1, B_2)$, $Z_4 = \text{SI-CDH}(U, Y)$. To solve the SI-CDH instance, $\mathcal{S}$ checks if there is an H query made by $\mathcal{M}$ of the form $(\Pi, Z_1, Z_2, Z_3, Z_4, \hat{A}, \hat{B}, U, Y)$, such that $\text{SI-DDH}(A_1, Y, Z_1) = 1$, $\text{SI-DDH}(U, B_2, Z_2) = 1$, $\text{SI-DDH}(A_1, B_2, Z_3) = 1$, and $\text{SI-DDH}(U, Y, Z_4) = 1$. If such an H query exists, $\mathcal{S}$ outputs Z_2 as the SI-CDH answer where $Z_2 = \text{SI-CDH}(U, B_2) = \text{SI-CDH}(U, V)$. With probability at least $\frac{1}{sn}$, the test session is $\mathtt{sid}^*$ with peer $\hat{B}$. Thus the advantage of $\mathcal{S}$ is

$$\mathbf{Adv}^{\text{di-SI-GDH}}_{\mathfrak{g},\mathfrak{e}}(\mathcal{S}) \geq \frac{1}{sn} \Pr[\mathsf{M} \cap \mathsf{H} \cap \mathsf{E}_1 \cap \overline{\mathsf{R}}]. \tag{2}$$

Notice that in the above simulation $\mathcal{S}$ cannot respond to $\mathsf{StaticKeyReveal}(\hat{B})$ query. However, given that event E_1 occurs, $\mathcal{S}$ correctly guesses the test session and the test session is fresh at the end of the experiment, then $\mathcal{M}$ have not queried for the static secret keys of the test session $\hat{B}$.

Such static key reveal query would contradict the freshness of the test session and thus the simulation terminated without errors.

$\mathsf{E}_1 \cap \mathsf{R}$. $\mathcal{S}$ prepares n honest parties, selects an honest party $\hat{A}$ and assigns its static public keys as $A_1 = U$ and $A_2 = V$. The remaining $n - 1$ parties are assigned random static and secret key pairs. $\mathcal{S}$ simulates the environment of $\mathcal{M}$ by following the protocol description. The party $\hat{A}$ is simulated as $\hat{B}$ in $\mathsf{E}_1 \cap \overline{\mathsf{R}}$.

If $\mathcal{M}$ selected a session whose owner and peer are the same party $\hat{A}$ as the test session, and $\mathsf{E}_1 \cap \mathsf{R}$ occurs, this simulation does not fail. Let $(\Pi, \mathcal{I}, \hat{A}, \hat{A}, X, Y)$ be the session identifier of the test session. When $\mathcal{M}$ is successful, $\mathcal{S}$ checks if there is an H query made by $\mathcal{M}$ of the form $(\Pi, Z_1, Z_2, Z_3, Z_4, \hat{A}, \hat{A}, X, Y)$, such that $\text{SI-DDH}(A_1, Y, Z_1) = 1$, $\text{SI-DDH}(X, A_2, Z_2) = 1$, $\text{SI-DDH}(A_1, A_2, Z_3) = 1$, and $\text{SI-DDH}(X, Y, Z_4) = 1$. If such an H query exists, $\mathcal{S}$ outputs Z_3 as the SI-CDH answer where $Z_2 = \text{SI-CDH}(A_1, A_2) = \text{SI-CDH}(U, V)$. With probability at least $\frac{1}{n}$, $\mathcal{M}$ will select a test session whose owner and peer is the same party $\hat{A}$. Thus, the advantage of $\mathcal{S}$ is

$$\mathbf{Adv}^{\text{di-SI-GDH}}_{\mathfrak{g},\mathfrak{e}}(\mathcal{S}) \geq \frac{1}{n} \Pr[\mathsf{M} \cap \mathsf{H} \cap \mathsf{E}_1 \cap \mathsf{R}]. \tag{3}$$

$\mathcal{S}$ cannot respond to $\mathsf{StaticKeyReveal}(\hat{A})$ query during the simulation. As before if event E_1 occurs, $\mathcal{S}$ correctly guesses the test session and the test session is fresh at the end of the experiment, then $\mathcal{M}$ have not queried for the static secret keys of $\hat{A}$, and therefore the simulation does not terminate with error.

$\mathsf{E}_2 \cap \overline{\mathsf{R}}$. $\mathcal{S}$ prepares n honest parties, selects two distinct honest parties $\hat{A}$ and $\hat{B}$, and assigns $\hat{A}$'s and $\hat{B}$'s static public keys as $A_1 = U$ and $B_2 = V$, respectively. $\mathcal{S}$ assigns random static public and secret key pairs for A_2, B_1, and the remaining $n-2$ parties are assigned random static and secret key pairs. $\mathcal{S}$ follows the protocol description when $\mathcal{M}$ activates session between honest peers, and simulate $\mathcal{M}$'s queries related to $\hat{A}$ or $\hat{B}$ as explained in E_1.

If $\mathcal{M}$ selected a session whose participants are $\hat{A}$, $\hat{B}$ as the test session, and $\mathsf{E}_2 \cap \overline{\mathsf{R}}$ occurs, this simulation does not fail. Let $(\Pi, \mathcal{I}, \hat{A}, \hat{B}, X, Y)$ be the session identifier of the test session. Note that $\mathcal{S}$ generated X and so knows $\mathfrak{x}$. When $\mathcal{M}$ is successful, $\mathcal{S}$ checks if there is an H query made by $\mathcal{M}$ of the form $(\Pi, Z_1, Z_2, Z_3, Z_4, \hat{A}, \hat{B}, X, Y)$, such that SI-DDH$(A_1, Y, Z_1) = 1$, SI-DDH$(X, B_2, Z_2) = 1$, SI-DDH$(A_1, B_2, Z_3) = 1$, and SI-DDH$(X, Y, Z_4) = 1$. If such an H query exists, $\mathcal{S}$ outputs Z_3 as the SI-CDH answer where $Z_3 = $ SI-CDH$(A_1, B_2) = $ SI-CDH(U, V). With probability at least $\frac{1}{n^2}$, $\mathcal{M}$ will select a test session with owner $\hat{A}$ and peer $\hat{B}$, respectively. Thus, the advantage of $\mathcal{S}$ is

$$\mathbf{Adv}^{\text{di-SI-GDH}}_{\mathfrak{g},\mathfrak{e}}(\mathcal{S}) \geq \frac{1}{n^2} \Pr[\mathsf{M} \cap \mathsf{H} \cap \mathsf{E}_2 \cap \overline{\mathsf{R}}]. \tag{4}$$

$\mathcal{S}$ can respond to neither $\mathsf{StaticKeyReveal}(\hat{A})$ nor $\mathsf{StaticKeyReveal}(\hat{B})$ queries during the simulation. As before if event E_2 occurs, $\mathcal{S}$ correctly guesses the test session and the test session is fresh at the end of the experiment, then $\mathcal{M}$ have queried for the static secret key of neither $\hat{A}$ nor $\hat{B}$, and therefore the simulation does not terminate with error.

$\mathsf{E}_2 \cap \mathsf{R}$. This case is essentially the same as $\mathsf{E}_1 \cap \mathsf{R}$. In this case, we can construct an di-SI-GDH solver $\mathcal{S}$ with advantage

$$\mathbf{Adv}^{\text{di-SI-GDH}}_{\mathfrak{g},\mathfrak{e}}(\mathcal{S}) \geq \frac{1}{n} \Pr[\mathsf{M} \cap \mathsf{H} \cap \mathsf{E}_2 \cap \mathsf{R}]. \tag{5}$$

E_3. $\mathcal{S}$ prepares n honest parties, and assigns random static public and secret key pairs for these parties. $\mathcal{S}$ selects $i, j \in_R \{1, \ldots, s\}$, and chooses i-th session $\mathtt{sid}^*$ and j-th session $\overline{\mathtt{sid}^*}$ among sessions activated by $\mathcal{M}$ and owned by honest parties. When activated, $\mathcal{S}$ sets the ephemeral public key of $\mathtt{sid}^*$ to be U and of $\overline{\mathtt{sid}^*}$ to be V. Since $\mathcal{S}$ knows the static secret keys of all honest parties, it can respond all queries, faithfully, except those that related to $\mathtt{sid}^*$ and $\overline{\mathtt{sid}^*}$.

Provided that $\mathcal{M}$ selects $\mathtt{sid}^*$ as the test session, $\overline{\mathtt{sid}^*}$ as its matching session, and E_3 occurs, the simulation does not fail. Let $(\Pi, \mathcal{I}, \hat{A}, \hat{B}, U, V)$ and $(\Pi, \mathcal{R}, \hat{B}, \hat{A}, U, V)$ be the session identifiers of $\mathtt{sid}^*$ and $\overline{\mathtt{sid}^*}$, respectively. When $\mathcal{M}$ wins the security game, $\mathcal{S}$ checks if there is an H query made by $\mathcal{M}$ of the form $(\Pi, Z_1, Z_2, Z_3, Z_4, \hat{A}, \hat{B}, U, V)$, such that SI-DDH$(A_1, V, Z_1) = 1$, SI-DDH$(U, B_2, Z_2) = 1$, SI-DDH$(A_1, B_2, Z_3) = 1$, and SI-DDH$(U, V, Z_4) = 1$. If such an H query exists, $\mathcal{S}$ outputs Z_4 as the SI-CDH answer. With probability at least $\frac{1}{s^2}$, $\mathcal{M}$ selects $\mathtt{sid}^*$ as the test session and $\overline{\mathtt{sid}^*}$ as its matching session. Thus the advantage of $\mathcal{S}$ is

$$\mathbf{Adv}^{\text{di-SI-GDH}}_{\mathfrak{g},\mathfrak{e}}(\mathcal{S}) \geq \frac{1}{s^2} \Pr[\mathsf{M} \cap \mathsf{H} \cap \mathsf{E}_3]. \tag{6}$$

$\mathcal{S}$ cannot respond to SessionStateReveal queries against the test session and its matching during the simulation. However, under event E_3 adversary does not issue such queries, and hence the simulation does not fail.

E_4, E_5, and E_6. The analysis of E_4, E_5, and E_6 is similar to E_2, E_1, and E_1, respectively. We omit the details and provide only the conclusion. In each case, we can construct an di-SI-GDH solver $\mathcal{S}$ as follows.

$$\mathbf{Adv}^{\text{di-SI-GDH}}_{\mathfrak{g},\mathfrak{e}}(\mathcal{S}) \geq \tfrac{1}{n^2} \Pr[\mathsf{M} \cap \mathsf{H} \cap \mathsf{E}_4 \cap \overline{\mathsf{R}}], \tag{7}$$

$$\mathbf{Adv}^{\text{di-SI-GDH}}_{\mathfrak{g},\mathfrak{e}}(\mathcal{S}) \geq \tfrac{1}{sn} \Pr[\mathsf{M} \cap \mathsf{H} \cap \mathsf{E}_i \cap \overline{\mathsf{R}}], \tag{8}$$

$$\mathbf{Adv}^{\text{di-SI-GDH}}_{\mathfrak{g},\mathfrak{e}}(\mathcal{S}) \geq \tfrac{1}{n} \Pr[\mathsf{M} \cap \mathsf{H} \cap \mathsf{E}_j \cap \mathsf{R}], \tag{9}$$

for $i = 5, 6$, and $j = 4, 5, 6$.

Analysis. Combining (1), ..., (9), we have

$$\mathbf{Adv}^{\text{di-SI-GDH}}_{\mathfrak{g},\mathfrak{e}}(\mathcal{S}) \geq \min\left\{\frac{1}{sn}, \frac{1}{n^2}, \frac{1}{s^2}\right\} \cdot \mathbf{Adv}^{\text{AKE}}_{\text{BCSIDH}}(\mathcal{M}).$$

During the simulation, the solvers $\mathcal{S}$ and $\mathcal{S}$ perform $\mathcal{O}\big((n+s)\lambda\big)$ low-degree isogeny operations for assigning static and ephemeral keys, and make $\mathcal{O}(h+s)$ times SI-DDH oracle queries for simulating SessionKeyReveal and the random oracle H queries. This completes the proof of Theorem 2. □

C Basic Facts on Elliptic Curves and Isogenies

We summarize basic facts and notations about elliptic curves and isogenies. For details, refer to [29, 14]. Let p be a prime greater than 3 and $\mathbb{F}_p$ be the finite field with p elements. Let $\overline{\mathbb{F}}_p$ be its algebraic closure. An elliptic curve E over $\overline{\mathbb{F}}_p$ is given by the Weierstrass normal form

$$E : Y^2 = X^3 + \alpha X + \beta \tag{10}$$

for α and $\beta \in \overline{\mathbb{F}}_p$ where the discriminant of the right hand side of Eq. (10) is non-zero. We denote the point at infinity on E by O_E. Elliptic curves are endowed with a unique algebraic group structure, with O_E as neutral element. The j-invariant of E is $j(E) = j(\alpha, \beta) = 1728\frac{4\alpha^3}{4\alpha^3+27\beta^2}$. Conversely, for $j \neq 0, 1728 \in \overline{\mathbb{F}}_p$, set $\alpha = \alpha(j) = \frac{3j}{1728-j}$, $\beta = \beta(j) = \frac{2j}{1728-j}$. Then, the obtained E in Eq. (10) has j-invariant j. Two elliptic curves over $\overline{\mathbb{F}}_p$ are isomorphic if and only if they have the same j-invariant. For a positive integer n, the set of n-torsion points of E is $E[n] = \{P \in E(\overline{\mathbb{F}}_p) \,|\, nP = O_E\}$.

Given two elliptic curves E and $\tilde{E}$ over $\overline{\mathbb{F}}_p$, a homomorphism $\phi : E \to \tilde{E}$ is a morphism of algebraic curves that sends O_E to $O_{\tilde{E}}$. A non-zero homomorphism is called an isogeny, and a separable isogeny with the cardinality ℓ of the kernel is called ℓ-isogeny. We consider only *separable* isogenies in this paper, i.e., any

isogeny is separable here. An elliptic curve E over $\overline{\mathbb{F}}_p$ is called supersingular if there are no points of order p, i.e., $E[p] = \{O_E\}$. The j-invariants of supersingular elliptic curves lie in $\mathbb{F}_{p^2}$ [29]. A non-supersingular elliptic curve is called ordinary.

We compute the ℓ-isogeny by using Vélu's formulas for a small prime $\ell = 2, 3, \ldots$. Vélu gave in [33] the explicit formulas of the isogeny $\psi : E \to \tilde{E}$ and the equation of $\tilde{E}$ when E is given by Eq. (10) and $\mathcal{K} = \ker \psi$ is explicitly given. Then there exists a unique isogeny $\psi : E \to \tilde{E}$ s.t. $\mathcal{K} = \ker \psi$, and we denote $\tilde{E}$ by $E/\mathcal{K}$. For an elliptic curve E and a cyclic group $\mathcal{K}(\subset E)$ of order ℓ, Vélu's formula [33] gives an isogenous curve $E/\mathcal{K}$ and the associated isogeny $E \ni (x, y) \mapsto (\tilde{x}, \tilde{y}) \in E/\mathcal{K}$. For computing it, for $E : Y^2 = X^3 + \alpha X + \beta$ and point $Q = (x_Q, y_Q) \neq O_E \in \mathcal{K}$, we define $g_Q^x = 3x_Q^2 + \alpha, g_Q^y = -2y_Q$, and $t_Q = 2g_Q^x$ if $Q \in E[2]$, $t_Q = g_Q^x$ if $Q \notin E[2]$, $u_Q = (g_Q^y)^2$. For $S = (\mathcal{K} - \{O_E\})/\pm 1$, let $t = \sum_{Q \in S} t_Q, w = \sum_{Q \in S}(u_Q + x_Q t_Q), \tilde{\alpha} = \alpha - 5t, \tilde{\beta} = \beta - 7w$, then, $\tilde{E} = E/\mathcal{K} : Y^2 = X^3 + \tilde{\alpha}X + \tilde{\beta}, \tilde{x} = x + \sum_{Q \in S} \left(\frac{t_Q}{x - x_Q} + \frac{u_Q}{(x - x_Q)^2} \right), \tilde{y} = y - \sum_{Q \in S} \left(\frac{2u_Q y}{(x - x_Q)^3} + \frac{t_Q(y - y_Q) - g_Q^x g_Q^y}{(x - x_Q)^2} \right)$ gives the curve and isogeny.