Post-Quantum Signal Key Agreement with SIDH

Samuel Dobson and Steven D. Galbraith

Mathematics Department, University of Auckland, New Zealand.
samuel.dobson.nz@gmail.com, s.galbraith@auckland.ac.nz

March 3, 2022

Abstract

In the effort to transition cryptographic primitives and protocols to quantum-resistant alternatives, an interesting and useful challenge is found in the Signal protocol. The initial key agreement component of this protocol, called X3DH, has so far proved more subtle to replace—in part due to the unclear security model and properties the original protocol is designed for. This paper defines a formal security model for the original Signal protocol, in the context of the standard eCK and CK+ type models, which we call the Signal-adapted-CK model. We then propose a secure replacement for the Signal X3DH key exchange protocol based on SIDH, and provide a proof of security in the Signal-adapted-CK model, showing our protocol satisfies all security properties of the original Signal X3DH. We call this new protocol SI-X3DH. Our protocol refutes the claim of Brendel, Fischlin, Günther, Janson, & Stebila [Selected Areas in Cryptography (2020)] that SIDH cannot be used to construct a secure X3DH replacement due to adaptive attacks. Unlike the generic constructions proposed in the literature, our protocol achieves deniability without expensive machinery such as post-quantum ring signatures. It also benefits from the efficiency of SIDH as a key-exchange protocol, compared to other post-quantum key exchange protocols such as CSIDH.

1 Introduction

Signal is a widely-used secure messaging protocol with implementations in its namesake app (Signal Private Messenger), as well as others including WhatsApp, Facebook Messenger and more. Due to its popularity, it is an interesting problem to design a post-quantum secure variant of the protocol. However, some difficulty arises due to the lack of a formally-defined security model or properties for the original protocol itself.

The Signal protocol consists of two general stages: the first is the initial key agreement, which is then followed by the double ratchet protocol [MP16a]. The initial key agreement is currently done via a protocol known as Extended Triple Diffie–Hellman (X3DH) [MP16b]. While Alwen, Coretti, and Dodis [ACD19] construct a version of the double ratchet component using key encapsulation mechanisms (KEMs), which can be made post-quantum secure, the X3DH stage has proven to be more subtle and challenging to replace in an efficient way with post-quantum solutions. Recent work by Brendel, Fischlin, Günther, Janson, and Stebila [BFG$^+$20] examines some of these challenges and suggests that SIDH cannot be used to make X3DH post-quantum secure due to its vulnerability to adaptive attacks when static keys are used.

Specifically, Brendel et al. are referring to an adaptive attack on SIDH given by Galbraith, Petit, Shani, and Ti [GPST16] (henceforth referred to as the GPST attack), which uses specially crafted points in a user's public key to extract bits of information about the isogeny path (and thus the secret key) of the other participant. The Signal X3DH protocol is an authenticated key exchange (AKE) protocol, requiring keys from both parties involved. Without a secure method of validating the correctness of the other party's keys, it would be insecure to perform a naive SIDH key exchange with them. For example, the initiator of a key exchange could adaptively modify the ephemeral public keys they use, in order to learn the receiver's long-term identity private key via this GPST attack.

Known methods of validation used to prevent adaptive attacks in SIDH are not well-suited to solving this issue in the Signal X3DH context. One proposed method of overcoming the GPST attack, known as k-SIDH [AJL17], has both parties use k different SIDH public keys, and runs k^2 instances of SIDH in parallel with pairwise combinations of these keys, combining all the shared secrets using a hash function in the final step of the protocol. The GPST attack was extended to k-SIDH in [DGL$^+$20] and shown to be feasible for small k (an attack on $k = 2$ is demonstrated concretely). Due to the possibility of attacking k-SIDH for small k, it has been suggested that k of at least 92 would be required to achieve security against quantum adversaries. Unfortunately, this makes the protocol very inefficient. An alternative which is commonly used, as in SIKE [CCH$^+$], is to convert the key exchange into a key encapsulation mechanism (KEM) using the Fujisaki–Okamoto (FO) transform or its variants [HHK17], and verify that the public key is well-formed and honestly generated [Pei14, KLM$^+$15]. The idea of the FO-transform is that the initiator, A, of the key exchange can encrypt the randomness they used in the exchange (for example, to generate their secret key) under the symmetric shared key K they derived, and send it to their partner B. If the encryption method is one-time secure, then because only A and B know K, only they can decrypt this randomness. B can then check that A performed the exchange protocol correctly, and in particular, that the public key they generated is indeed derived from the randomness they provided, to prove that A's public key is well-formed. Because B learns the secret key of A in every exchange, A can only do this with ephemeral keys. Hence, while extremely useful, the FO-transform does not provide a solution in cases where both parties use static keys. We cannot exclude the possibility that participants use their long-term (static) keys as part of an attack: a dedicated or well-resourced attacker could certainly register many new accounts whose identity keys are maliciously crafted, and initiate exchanges with an unsuspecting user (perhaps by marauding as their friends or colleagues) to learn their secret key. For these reasons, Brendel et al. disregard SIDH as a contender and suggest using CSIDH [CLM$^+$18] for an isogeny-based variant of Signal. However, this primitive is much less efficient than SIDH—in part due to sub-exponential quantum attacks that lead to much larger parameters.

One of the primary goals of this paper is to show that SIDH can indeed be used to construct a post-quantum X3DH replacement that satisfies the same security model as the original X3DH protocol—despite the claim by Brendel et al. [BFG$^+$20].

In order to design good post-quantum replacements for the Signal protocol, a clear security model is required. This is an area of difficulty because the original Signal protocol did not define a security model—it appears to be designed empirically. There have since been a few efforts to formalise the security properties of the Signal protocol and X3DH. Notably, the work by Cohn-Gordon, Cremers, Dowling, Garratt, and Stebila [CGCD$^+$20] was the first to propose a security model and prove the security of Signal in it. The recent work of Hashimoto, Katsumata, Kwiatkowski, and Prest [HKKP21] also proposes a generic security model for the Signal initial key agreement (specifically, for what they call Signal-conforming AKEs), and gives a generic construction from KEMs and signature schemes (as mentioned above, KEMs do not allow static–static key exchange, so a signature scheme is required to provide explicit authentication of the initiating party). From these analyses of the protocol, the following security properties have been identified as important, which any post-quantum replacement should therefore also satisfy:

1. Correctness: If Alice and Bob complete an exchange together, they should derive the same shared secret key.

2. Secrecy (also known as key-indistinguishability): Under the corruption of various combinations of the participants' secret keys, the shared secret for the session should not be recoverable, or even distinguishable from a random key. The combinations are defined by the specific security model used, for example, the CK model [CK01] or the model in [CGCD$^+$20]. This is, of course, a basic requirement for any secure key exchange.

3. (Implicit) authentication: Both participants should know who they are talking to, and be able to verify their partner's identity.

4. Perfect forward secrecy (PFS): Past communication should remain secure and unreadable by adversaries even if the participants' long-term keys are compromised in the future.

5. Asynchronicity: The protocol can be made non-interactive by hosting participants' public keys on a third-party server, which is untrusted. In the security model, the only possible malicious ability the server should have is that it could deny Alice the retrieval of Bob's keys (or, say, not give out his one-time keys). This property is also called *receiver obliviousness* by Hashimoto et al. [HKKP21], because the uploaded public keys are not intended for any particular user, but can be retrieved and used by anyone.

6. (Offline) deniability [VGIK20], also known as identity-hiding: The transcript of an exchange session should not reveal the participants of the exchange (in a non-repudiable way).

We propose a new, efficient, post-quantum key exchange protocol using SIDH, modelled after X3DH, which we call SI-X3DH. This new protocol solves the problem of adaptive attacks by using a variant of the FO transformation to prove that the initiator's ephemeral key is honestly generated, and a Proof of Knowledge to ensure the long term public keys are well-formed—something which only needs to be verified once (and could be offloaded to the PKI server depending on the trust model used). We prove security of the SI-X3DH protocol formally in the random oracle model (ROM) using a new key-indistinguishability model we call the Signal-adapted-CK model. We show the security of SI-X3DH reduces to the hardness of the supersingular isogeny CDH (SI-CDH) assumption in the ROM.

Because SIDH is an efficient post-quantum key exchange proposal with very small key sizes (although still larger than classical elliptic curve keys used in the original X3DH), and because SI-X3DH requires only three or four SIDH exchanges (unlike k-SIDH), our protocol is also efficient and practical. For example, SIDH is much faster than CSIDH—suggested in the proposal by Brendel et al. [BFG$^+$20]—because CSIDH uses larger-prime degree isogenies while SIDH commonly uses only isogenies of degree (a power of) two and three. Our scheme also does not rely on expensive machinery such as post-quantum ring signatures to achieve deniability (as [HKKP21] does). However, a large drawback of our scheme is that it relies on proving knowledge of the secret long-term identity keys, by using the SIDH Proof of Knowledge from [DDGZ21] for example. This only needs to be done once per contact (or could be offloaded to the keyserver, depending on the trust model), but for users who add many new contacts regularly, this may create an unacceptable overhead. The efficiency of our scheme is discussed more in Section 7.

Another disadvantage of our scheme, as discussed in Section 5, is that SI-X3DH suffers from the possibility of more permanent key compromise impersonation (KCI) than the original Signal X3DH protocol does. Technically, neither Signal X3DH nor SI-X3DH satisfy the KCI resistance requirement of the eCK and CK+ security models, but there is a practical difference between the schemes. Impersonation was possible with the compromise of the semi-static key in Signal X3DH, whereas in SI-X3DH, impersonation is possible with compromise of the long-term identity key. Thus, cycling the semi-static key is no longer sufficient to prevent long-term impersonation. This is worth considering, but we believe the change is acceptable, as medium-term impersonation seems just as damaging as long-term, and corruption of an identity key is a severe break in security anyway.

As we will soon see, the SI-X3DH protocol we propose has some structural differences from X3DH. In particular, SI-X3DH performs an SIDH exchange between the two parties' identity keys (IK_A and IK_B), whereas previously, X3DH used IK_A and SK_B instead (involving Bob's semi-static key, rather than his identity key). Due to the asymmetry between the degrees of the isogenies the two parties in SIDH use, our protocol requires parties to register two keys rather than one: a receiving key and a sending key. Finally, in order to prevent adaptive attacks, SI-X3DH uses a single FO-proof per exchange, and a once-off proof of well-formedness of each party's identity keys (see Section 5 for discussion of this). Despite these differences, the structure of the protocol more closely resembles X3DH than any of the other post-quantum proposals presented to date. For example, our protocol allows Bob the balance between one-time keys and medium-term (semi-static) keys—where the former may be exhausted, leading to denial of service, while the latter provide less security in some attack scenarios. These properties and differences are discussed further in Section 4.

1.1 Related work

Brendel et al. [BFG$^+$20] proposed a new model for post-quantum X3DH replacements using a primitive they call split-KEMs. Their construction is a theoretical work, as they leave it an open question whether post-quantum primitives such as CSIDH satisfy the security definitions of their split-KEM.

Recently, Hashimoto et al. [HKKP21] presented their Signal-Conforming AKE (SC-AKE) construction, also using post-quantum KEMs to construct a generic Signal X3DH replacement. To achieve deniability, their scheme requires a post-quantum ring signature scheme. Independently, but following a very similar approach to Hashimoto et al., Brendel et al. [BFG$^+$22] also proposed a deniable AKE using post-quantum KEMs (which they call "Signal in a Post-Quantum Regime" (SPQR)) and a designated verifier signature (DVS) scheme. As they mention, little work has been done to date in constructing DVS schemes from post-quantum assumptions, so Brendel et al. also propose using a two-party post-quantum ring signature scheme for the same purpose.

We briefly outline the differences between these works and that presented in this chapter using Table 1, with the original Signal X3DH protocol included as a reference.

Scheme	PQ-secure	Deniable	Requires sig	Long-term data	Exchanged data
Original Signal X3DH protocol	×	✓	✓	K	3 keys
Split-KEM based X3DH [BFG$^+$20]	✓	?	✓	K, K_σ	3 keys, 4 ciphertexts
Signal-Conforming AKE [HKKP21]	✓	*with PQ ring signature	✓(×2)	K, K_σ, K_σ^*	1 key, 3 ciphertexts
SPQR [BFG$^+$22]	✓	*with PQ ring signature or DVS	✓(×2)	K, K_σ, K_σ^*	2 keys, 4 ciphertexts
SI-X3DH (this work)	✓	✓	✓	K_2, K_3, K_σ + PoK	3 keys, 1 ciphertext

Table 1: Comparison of post-quantum Signal X3DH replacements. *Long-term data* refers to the size of the initial registration cost for each user (the "offline" data). *Exchanged data* gives the amount of ephemeral data sent in a single exchange (by both parties combined), that is, the size of the "online" transcript. Note that all schemes require a signature scheme (*Requires sig*) to obtain PFS—post quantum schemes use a separate signature verification key K_σ while Signal X3DH reused the same key K for both exchange and signature verification (ECDH and XEdDSA [Per16]).

The original Signal X3DH scheme requires Bob to sign his semi-static keys, to prevent a malicious keyserver from providing its own keys and compromising the perfect forward secrecy guarantee of the scheme. This requirement must still hold in any post-quantum replacement too. The Split-KEM protocol [BFG$^+$20] does not discuss the requirement for a signature scheme on the semi-static keys, but the same attack on PFS applies to their scheme as it does to the original Signal X3DH protocol if the semi-static keys are not signed—a malicious server or tampering man-in-the-middle can insert their own semi-static key rather than Bob's, and later compromise Bob's long-term identity key, thus allowing recovery of the shared secret. The Signal-Conforming AKE protocol and SPQR protocol require this signature for PFS too, for the same reason. These latter two schemes also use a second (ring/DVS) signature (discussed below)—two signatures per exchange. Because ring signatures and DVS schemes are much more expensive than standard signatures, for efficiency it would likely be preferable to use two separate schemes, hence the two signing keys K_σ, K_σ^* in Table 1. Our construction, as mentioned above, requires a single signature on the semi-static key. Because there are no efficient post-quantum constructions with a public key that can be used in both a signature scheme and a key exchange, requiring a separate signature scheme (and verification key) seems unavoidable for any post-quantum X3DH replacement. In general, these X3DH replacements (including SI-X3DH) are agnostic to the signature scheme used for this purpose, so any efficient post-quantum signature scheme may be used alongside them—there is no restriction to use an isogeny-based signature scheme with SI-X3DH.

For deniability, SC-AKE requires the initiator of the key exchange to sign the session ID. This signature creates non-repudiable evidence of the initiator's involvement in the exchange. Hashimoto et al. [HKKP21] and Brendel et al. [BFG$^+$22] suggest using a ring signature to attain deniability. Specifically, a signature under a two-party ring involving just the sender and receiver is sufficient to authenticate the other party in the exchange (since one party knows the signatures that they themselves generated), but to a third party, the signature could have been generated by either participant. Unfortunately, however, a post-quantum ring signature scheme is a much more expensive construction than a standard signature. Deniability of the split-KEM construction is not discussed by the 2020 work of Brendel et al. [BFG$^+$20], and would appear to depend on how the split-KEM is instantiated. We emphasise that the signature on Bob's semi-static keys mentioned above does not have any impact on deniability, as that signature exists independently of any particular exchange session or counterparty. These deniability drawbacks are only caused by signatures on session-specific information like the session ID, for the sake of authentication.

Finally, it is important to note that the SC-AKE protocol does not use a semi-static key—only long-term and ephemeral keys. This means that unlike in Signal X3DH, if a receiver is offline for an extended period of time, it is possible for all the ephemeral keys they uploaded to the server to be exhausted (either due to popularity or a malicious attempt to do so). This creates an opportunity for denial of service which is not present when semi-static keys are used and the ephemeral component is optional. Brendel et al. [BFG$^+$22] address this by using a semi-static and an ephemeral KEM encapsulation key if available, as in Signal's X3DH.

In other recent work, Fouotsa and Petit [FP21] propose a protocol similar to SIDH which they claim is not vulnerable to adaptive attacks. They call this protocol HealSIDH ("healed" SIDH). This protocol operates by requiring participants to also reveal the action of their isogenies on points of larger order than in SIDH. However, this protocol is interactive and would not allow a key exchange to take place while one participant is offline—it requires the receiver to send certain points back to the initiator for validation before the exchange can be completed. Specifically, this fails the requirement of asynchronicity, so would not be suitable for use in a Signal X3DH replacement. It is for the same reason that proposals for post-quantum TLS handshake replacements, including by Schwabe, Stebila, and Wiggers [SSW20], also fail to be applicable in the Signal context—these protocols involve messages sent by both parties sequentially over multiple rounds, and often do not authenticate one of the two parties (the client).

1.2 Outline

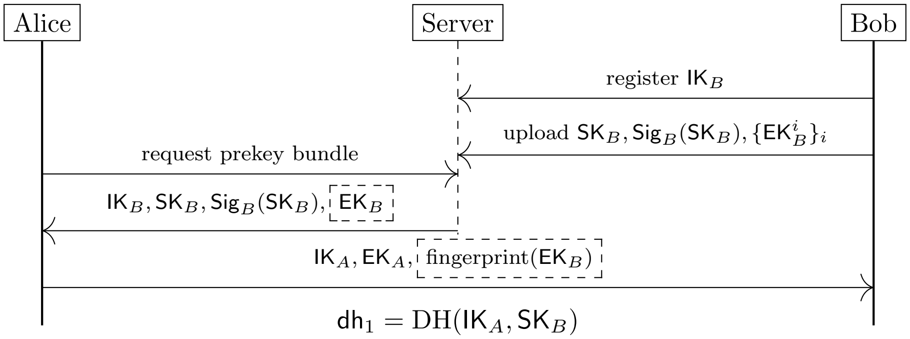
We shall begin in Section 2 by reviewing the existing X3DH protocol used as Signal's initial key agreement. We will then review the supersingular isogeny Diffie–Hellman key exchange (SIDH) in Section 3. In Section 4 we shall discuss the security properties of an appropriate Signal key agreement protocol in more detail and define a security model to be used. This is followed by our construction of a new protocol in Section 5 using SIDH, which we propose is an efficient post-quantum replacement for X3DH. Section 6 gives a proof of security for this construction, and Section 7 discusses the efficiency of our protocol and the key differences between our proposal and the original X3DH scheme.

1.3 Acknowledgements

We thank the anonymous reviewers for their helpful comments and feedback. We also thank Jason LeGrow for his feedback and advice.

2 The Signal X3DH protocol

The basic process of the X3DH protocol is given in Figure 1, where Alice is the initiator and Bob is the responder. Let $\mathrm{DH}_{\mathsf{pp}}(g^a, g^b) = g^{ab} \pmod N$ denote the result of a Diffie–Hellman key exchange between keys A and B (at least one of the private keys is needed to compute this, but the result is unambiguous), with public parameters pp including g and N. Because we assume fixed public parameters, we will usually omit the subscript. Throughout this paper, we will use $\boxed{\text{dashed boxes}}$ to denote optional parameters which may be omitted.


Alice
Server
Bob
register IK_B
upload $\mathsf{SK}_B, \mathsf{Sig}_B(\mathsf{SK}_B), \{\mathsf{EK}_B^i\}_i$
request prekey bundle
$\mathsf{IK}_B, \mathsf{SK}_B, \mathsf{Sig}_B(\mathsf{SK}_B), \boxed{\mathsf{EK}_B}$
$\mathsf{IK}_A, \mathsf{EK}_A, \boxed{\text{fingerprint}(\mathsf{EK}_B)}$

$$\mathsf{dh}_1 = \mathrm{DH}(\mathsf{IK}_A, \mathsf{SK}_B)$$
$$\mathsf{dh}_2 = \mathrm{DH}(\mathsf{EK}_A, \mathsf{IK}_B)$$
$$\mathsf{dh}_3 = \mathrm{DH}(\mathsf{EK}_A, \mathsf{SK}_B)$$
$$\boxed{\mathsf{dh}_4 = \mathrm{DH}(\mathsf{EK}_A, \mathsf{EK}_B)}$$
$$K = \mathrm{KDF}(\mathsf{dh}_1 \parallel \mathsf{dh}_2 \parallel \mathsf{dh}_3 \boxed{\parallel \mathsf{dh}_4})$$

Figure 1: The X3DH protocol [MP16b]. dh_4 is optional on the basis of one-time key availability.

Because the X3DH protocol is designed to work when the recipient (Bob) is offline, Alice obtains his public key information from a server. IK_A and IK_B are the fixed long-term identity keys of Alice and Bob respectively. Bob additionally uploads a semi-static public key SK_B signed by him to the server, which he rotates semi-regularly. He also uploads a number of one-time keys EK_B, but the use of these is optional as the supply on the server may run out.

After Alice has received Bob's identity, semi-static, and (optional) one-time keys from the server, she performs a three- or four-part key exchange with her own identity key and ephemeral key. These three or four shared keys are specified in the figure (denoted by dh_i), and are combined using some sort of secure hash or key derivation function (KDF). We shall assume they are simply concatenated and hashed with a cryptographic hash function. This results in the master shared secret for the exchange, which is then used in subsequent protocols such as Signal's Double Ratchet protocol.

Finally, Alice sends to Bob identifiers of which of his semi-static and one-time public keys she used (for example, short fingerprint), as well as her own identity and ephemeral keys. This allows Bob to also compute the same shared master secret.

Verification of the long-term identity keys is out-of-scope for the protocol, and may be done either by trusting a third party (e.g. the server) as a PKI, or verifying the keys in-person or out-of-band in some other way.

3 SIDH

We now provide a brief refresher on the Supersingular Isogeny Diffie–Hellman (SIDH) key exchange protocol [JDF11, DFJP14] by De Feo, Jao, and Plût.

As public parameters, we have a prime $p = \ell_1^{e_1}\ell_2^{e_2}f \pm 1$, where ℓ_1, ℓ_2 are distinct small primes, f is an integer cofactor, and $\ell_1^{e_1} \approx \ell_2^{e_2}$. We work over the finite field $\mathbb{F}_{p^2}$. Additionally we fix a base supersingular elliptic curve E_0 and a basis $\{P_i, Q_i\}$ for both the ℓ_1 and ℓ_2 torsion subgroups of $E(\mathbb{F}_{p^2})$ (such that $E_0[\ell^{e_i}] = \langle P_i, Q_i\rangle$). Typically $\ell_1 = 2$ and $\ell_2 = 3$, and this will be assumed from here forward in this paper. We will use both the index 1 and the subscript A to represent Alice's information, while $B \simeq 2$ will be used interchangeably for Bob's, for clarity in various situations and for consistency with existing literature.

It is well known that knowledge of an isogeny (up to equivalence, i.e., post-composition with an isomorphism) and knowledge of its kernel are equivalent, and we can convert between them at will, via Vélu's formulae [Vél71]. In SIDH, the secret key of Alice (respectively Bob) is an isogeny $\phi : E(\mathbb{F}_{p^2}) \to E_A(\mathbb{F}_{p^2})$ of degree 2^{e_1} (respectively 3^{e_2}). These isogenies are generated by randomly choosing a secret integer $\alpha \in \mathcal{K}_A$ and computing the isogeny whose kernel $K = \langle P_A + [\alpha]Q_A\rangle$[1]. We thus unambiguously refer to the isogeny, its kernel, and such an integer α, as "the secret key." Figure 2 depicts the commutative diagram making up the SIDH key exchange.

$$\begin{tikzcd}
E \arrow[r, "\phi_A"] \arrow[d, "\phi_B"'] & E_A \arrow[d, "\phi_{AB}"] \\
E_B \arrow[r, "\phi_{BA}"'] & E_{AB}
\end{tikzcd}$$

Figure 2: Commutative diagram of SIDH, where $\ker(\phi_{BA}) = \phi_B(\ker(\phi_A))$ and $\ker(\phi_{AB}) = \phi_A(\ker(\phi_B))$.

In order to make the diagram commute, Alice and Bob are required to not just give their image curves E_A and E_B in their respective public keys, but also the images of the basis points of the other participant's kernel on E. That is, Alice provides E_A, $P'_B = \phi_A(P_B), Q'_B = \phi_A(Q_B)$ as her public key. This allows Bob to "transport" his secret isogeny to E_A and compute ϕ_{AB} whose kernel is $\langle P'_B + [\beta]Q'_B\rangle$. Both Alice and Bob will arrive along these transported isogenies at distinct, but isomorphic, image curves E_{AB}, E_{BA}. Two elliptic curves are isomorphic over $\overline{\mathbb{F}}_{p^2}$ if and only if their j-invariants $j(E_{AB}) = j(E_{BA})$, hence this j-invariant may be used as the shared secret of the SIDH key exchange.

Throughout this paper, we will use the function $\mathrm{SIDH}_{\mathsf{pp}}(\cdot,\cdot)$ to represent this protocol with respect to public parameters pp, outputting the final j-invariant. Generally, the public parameters will be clear from context, so they may be omitted for ease of notation. The arguments to SIDH will be the two public keys of the participants, because clearly the result is independent of which participant computed the value (using their secret key). Specifically, if β is the secret key corresponding to the public key $K_B = (E_B, P'_A, Q'_A)$, then $\mathrm{SIDH}_{\mathsf{pp}}((E_A, P'_B, Q'_B), K_B) = j(E_A/\langle P'_B + [\beta]Q'_B\rangle)$.

3.1 SIDH assumptions

Let pp denote the public parameters $\mathsf{pp} = (p, \ell_1, \ell_2, e_1, e_2, E_0, P_1, Q_1, P_2, Q_2)$. The standard computational and decisional hardness assumptions associated with the SIDH key exchange are as follows. Let

$$\mathrm{SSEC}_{\mathsf{pp},i} = \{(E_i,\ \phi_i(P_{3-i}),\ \phi(Q_{3-i})) \mid \phi_i : E_0 \to E_i,\ \deg\phi_i = \ell_i^{e_i}\}$$

[1] This uses the idea of equivalent keys from Galbraith et al. [GPST16], and only uses keys of the form $(1, \alpha)$, of which there are 2^{e_1} and 3^{e_2} respectively. Restricting to such keys is standard in SIDH-based schemes, including SIKE.

be the set of all possible public keys for participant i in the SIDH protocol with public parameters pp. Let

$$\mathrm{SSJ}_{\mathsf{pp}} = \{j(E_i) \,:\, E_i \text{ defined over } \mathbb{F}_q \text{ and supersingular}\}$$

be the set of all supersingular j-invariants over the field $\mathbb{F}_q$ established by the public parameters pp. Every shared secret arising from an SIDH key exchange with public parameters pp is therefore contained in this set.

Definition 1 (Computational Supersingular-Isogeny Diffie–Hellman (SI-CDH) Problem.)**.** *Given the public parameters* pp*, and two public keys* $K_1 = (E_1, P_1', Q_1') \in \mathrm{SSEC}_{\mathsf{pp},1}$*,* $K_2 = (E_2, P_2', Q_2') \in \mathrm{SSEC}_{\mathsf{pp},2}$*, compute the* j*-invariant* $j = j(E_{12}) = j(E_{21}) = \mathrm{SIDH}_{\mathsf{pp}}(K_1, K_2)$*.*

We define the advantage of a probabilistic polynomial-time (PPT) adversary $\mathcal{A}$ solving the SI-CDH problem as

$$\mathsf{Adv}^{\text{si-cdh}}(\mathcal{A}) = \Pr\left[\, j' = \mathrm{SIDH}_{\mathsf{pp}}(K_1, K_2) \mid j' \leftarrow \mathcal{A}(\mathsf{pp}, K_1, K_2) \,\right].$$

The SI-CDH assumption states that for any PPT adversary $\mathcal{A}$, $\mathsf{Adv}^{\text{si-cdh}}(\mathcal{A}) \leq \mathsf{negl}$. In other words, given the two keys involved in an SIDH exchange, it should be infeasible to compute the resulting shared secret of the exchange.

Definition 2 (Decisional Supersingular-Isogeny Diffie–Hellman (SI-DDH) Problem.)**.** *Let* pp *be SIDH public parameters. Define two distributions:*

- $\mathcal{D}_0 = \{(K_1, K_2, j) \mid K_i \in \mathrm{SSEC}_{\mathsf{pp},i},\ j = \mathrm{SIDH}_{\mathsf{pp}}(K_1, K_2)\}$,
- $\mathcal{D}_1 = \{(K_1, K_2, j) \mid K_i \in \mathrm{SSEC}_{\mathsf{pp},i},\ j \leftarrow \mathrm{SSJ}_{\mathsf{pp}}\}$,

The SI-DDH problem is to distinguish between the distributions $\mathcal{D}_0$ *and* $\mathcal{D}_1$*.*

We define the advantage of a PPT adversary $\mathcal{A}$ solving the SI-DDH problem as

$$\mathsf{Adv}^{\text{si-ddh}}(\mathcal{A}) = \left| \Pr\left[\, b = b' \mid b' \leftarrow \mathcal{A}(\mathsf{pp}, K_1, K_2, j_b),\ b \leftarrow \{0,1\} \,\right] - \frac{1}{2} \right|,$$

where $(K_1, K_2, j_b) \in \mathcal{D}_b$. The SI-DDH assumption states that for any PPT adversary $\mathcal{A}$, $\mathsf{Adv}^{\text{si-ddh}}(\mathcal{A}) \leq \mathsf{negl}$.

3.2 New SI-CDH-based assumptions

We now present two new computational assumptions, both based on the standard SI-CDH problem from Definition 1. We sketch proofs that the SI-CDH problem can be reduced to both, in the random oracle model. These two assumptions are simply tools to simplify the proof of security of our new SI-X3DH protocol.

As usual, let pp be fixed SIDH public parameters. For ease of notation, let $\mathcal{K}_i$ (the i-keyspace) be the set of possible isogenies of degree $\ell_i^{e_i}$ from the fixed SIDH base curve E_0. Equivalently, $\mathcal{K}_i$ is the set of points of exact order $\ell_i^{e_i}$ on E_0, acting as isogeny kernel generators, where two generators are identified as the same key if they generate the same kernel. Let $H_1 : \{0,1\}^* \to \mathcal{K}_i$ be a pseudorandom generator (PRG) whose codomain is this secret isogeny keyspace. We also let $H_2 : \{0,1\}^* \to \{0,1\}^n$ be a PRG. Both H_1 and H_2 are modelled as random oracles. $\mathsf{PubkeyFromSecret}$ is a function taking a secret isogeny or kernel generator and outputting the codomain curve of that isogeny (or the isogeny with that kernel, via Vélu's formulae).

The first new SI-CDH-type assumption we define adds a "check" oracle to the SI-CDH assumption, which is provided by the challenge generator. This lets the adversary "verify" their answer before returning it to the challenger, so we call this the **Verifiable SI-CDH problem**.

Definition 3 (Verifiable SI-CDH (VCDH) problem)**.** *Let* pp *be SIDH public parameters, and* $K_1 \in \mathrm{SSEC}\,\mathsf{pp}, 1$ $K_2 \in \mathrm{SSEC}\,\mathsf{pp}, 2$ *be two SIDH public keys. Let* $\mathcal{O}_{K_1,K_2}$ *be a truth oracle defined as*

$$\mathcal{O}_{K_1,K_2}(j') = \begin{cases} 1 & \text{if } j' = \mathrm{SIDH}_{\mathsf{pp}}(K_1, K_2), \\ 0 & \text{otherwise.} \end{cases}$$

The Verifiable SI-CDH problem is to compute the j*-invariant* $j = \mathrm{SIDH}_{\mathsf{pp}}(K_1, K_2)$*, given* pp*,* K_1*,* K_2*, and* $\mathcal{O}_{K_1,K_2}$*.*

Essentially oracle $\mathcal{O}_{K_1,K_2}$ confirms if the answer to the SI-CDH challenge is correct or not. Ergo, intuitively we should learn no extra information from this oracle—on all except one j-invariant the oracle will return 0, so in polynomially-many queries, the likelihood of guessing the correct j-invariant is negligible (as in the SI-CDH problem). We show that, in the random oracle model, this problem is hard if the SI-CDH problem is.

Theorem 1. *Let* $\mathcal{B}$ *be an adversary solving the VCDH problem with advantage* ϵ *after making* q *queries to the oracle* $\mathcal{O}_{K_1,K_2}$*. Then* $\mathcal{B}$ *can be used to solve the SI-CDH problem with probability at least* $\epsilon/2q$*.*

Proof. Without loss of generality, we assume all q queries are made with distinct inputs. Let (K_1, K_2) be an SI-CDH challenge instance. We define two different oracles $\mathcal{O}^0$ and $\mathcal{O}^1$. Oracle $\mathcal{O}^0$ will return 0 regardless of the query made. To define oracle $\mathcal{O}^1$, we select a random index $0 \leq \ell < q$ and let $\mathcal{O}^2$ return 1 on the ℓ-th unique query (and 0 on all other queries). We run the adversary $\mathcal{B}$ in two settings, giving instance $(K_1, K_2, \mathcal{O}^i)$ to $\mathcal{B}$ in setting $i \in \{0, 1\}$. Define found to be the event that $\mathcal{B}$ makes a query to the oracle $\mathcal{O}$ it is given with the correct j-invariant (the solution to the SI-CDH instance). We can consider the probability of $\mathcal{B}$ succeeding against the VCDH problem as

$$\begin{aligned} \epsilon = & \Pr[\,\mathcal{B} \text{ wins} \mid \mathsf{found} \text{ occurs}\,] \cdot \Pr[\,\mathsf{found} \text{ occurs}\,] \\ & + \Pr[\,\mathcal{B} \text{ wins} \mid \mathsf{found} \text{ does not occur}\,] \cdot \Pr[\,\mathsf{found} \text{ does not occur}\,]. \end{aligned}$$

If found does not occur, then $\mathcal{B}$ running in setting 0 (where oracle $\mathcal{O}^0$ always returns 0) will be unable to distinguish the simulated oracle from the true one, and will win with advantage ϵ. Hence,

$$\Pr[\,\mathcal{B} \text{ wins in setting } 0\,] \geq \Pr[\,\mathcal{B} \text{ wins} \mid \mathsf{found} \text{ does not occur}\,].$$

On the other hand, if found occurs, then we correctly simulated the oracle in setting 1 with probability $1/q$ (the probability that we guessed ℓ correctly). Therefore,

$$\Pr[\,\mathcal{B} \text{ wins in setting } 1\,] \geq \frac{1}{q} \Pr[\,\mathcal{B} \text{ wins} \mid \mathsf{found} \text{ occurs}\,].$$

We uniformly sample $b \leftarrow \{0, 1\}$ and return the solution from $\mathcal{B}$ running in setting b to the SI-CDH challenger. Because $0 \leq \Pr[\,\mathsf{found} \text{ occurs}\,] \leq 1$, we solve the SI-CDH instance with overall probability

$$\begin{aligned} & \frac{1}{2} \Pr[\,\mathcal{B} \text{ wins in setting } 0\,] + \frac{1}{2} \Pr[\,\mathcal{B} \text{ wins in setting } 1\,] \\ \geq & \frac{1}{2} \Pr[\,\mathcal{B} \text{ wins} \mid \mathsf{found} \text{ does not occur}\,] + \frac{1}{2q} \Pr[\,\mathcal{B} \text{ wins} \mid \mathsf{found} \text{ occurs}\,] \\ \geq & \frac{1}{2q} \left(\Pr[\,\mathcal{B} \text{ wins} \mid \mathsf{found} \text{ does not occur}\,] + \Pr[\,\mathcal{B} \text{ wins} \mid \mathsf{found} \text{ occurs}\,]\right) \\ & \geq \frac{1}{2q}\epsilon, \end{aligned}$$

which is non-negligible if ϵ is (since q must be polynomially-sized). □

We call the second of our new SI-CDH-type problems the **Honest SI-CDH problem** (HCDH). This problem models an SI-CDH instance with an additional FO-like proof that the first key in the instance, K_1, was honestly generated.

Definition 4 (Honest SI-CDH (HCDH) problem)**.** *Let* pp *be SIDH public parameters, and* $s \leftarrow \{0,1\}^n$ *be a random seed, where* n *is the security parameter. Then, let*

$$K_1 = \mathsf{PubkeyFromSecret}(H_1(s))$$

be a public key derived from s*, where* $H_1(s)$ *is an isogeny of degree* $\ell_1^{e_1}$*. Let* $K_2 \in \mathrm{SSEC}_{\mathsf{pp},2}$ *be a second public key. Finally, let* π *be an FO-proof of the form*

$$\pi = s \oplus H_2(\mathrm{SIDH}_{\mathsf{pp}}(K_1, K_2)).$$

The Verifiable SI-CDH problem is, given pp*,* K_1*,* K_2*, and* π*, to compute the* j*-invariant* $j = \mathrm{SIDH}_{\mathsf{pp}}(K_1, K_2)$*.*

We argue that the FO-like proof leaks no information because we obviously assume that $\mathrm{SIDH}_{\mathsf{pp}}(K_1, K_2)$ is unknown (since it is the answer to the SI-CDH problem) and s is random. Thus, if the SI-CDH problem is hard, then so too is this problem. We sketch a reduction in the random oracle model. Treat H_2 as a random oracle. Let $\mathcal{B}$ be an adversary making q queries to H_2 and winning with advantage ϵ against the HCDH problem. Obtain an SI-CDH challenge (K_1, K_2). Choose π to be a random binary string, and provide (K_1, K_2, π) to $\mathcal{B}$.

In order to distinguish the simulated π from an honest FO-proof, $\mathcal{B}$ must query $H_2(j)$ for the correct j-invariant solution of the SI-CDH instance. If this occurs, we can return one of the q queries made to H_2 and win with probability $1/q$. Otherwise, the output of $\mathcal{B}$ wins with advantage ϵ despite π being uniformly random, by a simple hybrid argument.

Thus, the reduction can simply return one of the q queries to H_2 or the output of $\mathcal{B}$ to the SI-CDH challenger with equal probability. In either case, there is a non-negligible chance that the returned value wins the SI-CDH challenge, if ϵ is non-negligible.

4 Security model

Authenticated key exchange (AKE) security is a complex field of security properties and models. Of primary interest is the notion of key indistinguishability, sometimes simply known as AKE security due to its universality. The seminal work by Bellare and Rogaway [BR93] defined a security model for authenticated key exchange (known as the BR model). Security in the BR model is based on the indistinguishability of true session keys from random, even when the adversary is given certain powers to control protocol flow, interactions, and to reveal long-term secret keys and states. A number of other models have since been developed, based on this original BR model, including the CK [CK01], CK+ [Kra05] and eCK [LLM07] models. These models all differ based on the powers of the adversary in the key-indistinguishability game (as well as other differences such as how partner sessions and session IDs are defined). The main difference between the CK/CK+ models and the eCK model is that the latter uses ephemeral-key reveal queries while the former use session-state reveal queries. These models are incomparable [Cre09].

The eCK and CK+ models are generally viewed as the strongest and most desirable models, as they capture attacks which are outside the scope of the CK model: weak perfect forward secrecy (wPFS), key compromise impersonation (KCI), and maximal exposure (MEX). All of these properties relate to certain combinations of long-term and ephemeral keys being compromised by an adversary. Security in these models relies on allowing the adversary all non-trivial combinations of exposure—i.e. any combination of keys from both parties that does *not* form a vertex cover on the graph of Diffie–Hellman exchanges in the protocol (the graph whose nodes are keys, and edges represent that a DH key exchange between the two incident keys is used in the protocol). A vertex cover would trivially allow the adversary to compute the shared secret, because at least one secret is known to the adversary in every DH exchange (edge). But if the adversary does not have a vertex cover, at least one DH exchange cannot be computed, because the adversary does not have either of the secret keys involved. In this case, the overall session key of the protocol should remain hidden. We refer the reader to the work of Fujioka et al. [FSXY12] for a more detailed analysis of the difference between these models.

Unfortunately, Signal X3DH does not meet the definition of security required by all these models. This was observed by Cohn-Gordon et al. [CGCD$^+$20]. Precisely, there do not exist edges in the exchange graph for every possible pair of keys—for example, there is no DH exchange between Alice's identity key and Bob's identity or ephemeral keys. Our benchmark for security is that a replacement protocol should meet at least the same security definition as that of the original protocol, so we must observe where exactly the original protocol breaks down in the eCK model. This allows us to propose a slightly weaker model, though still stronger than the CK model, that successfully represents the security goals of Signal X3DH. This gives a more formal and well-defined security model than the one Cohn-Gordon et al. [CGCD$^+$20] used to prove security of the original Signal X3DH protocol. We call our new security model the Signal-adapted-CK model.

The recent work of Hashimoto et al. [HKKP21] provided a similar security model, for what they call a Signal-conforming AKE protocol. Their security model differs from ours in the fact that it does not take semi-static keys into account (their proposed construction does not use semi-static keys). They also use the language of state-reveals rather than ephemeral-key-reveals. Their model is stronger than the Signal-adapted-CK model—in fact, the original Signal X3DH protocol would not satisfy their model (it requires security against the two events E_4 and E_8 in Table 3, discussed further below). However, our goal is to propose a model that exactly captures the security properties of the original Signal X3DH protocol, which was not the goal of their model. In other words, we wish to analyse Signal, not some stronger protocol.

Before we begin, let us briefly recall the meanings of the security notions mentioned above:

- Perfect forward secrecy (PFS) implies that an adversary who corrupts one or both of the participants' long-term secret keys should not be able to reveal the session key of previous sessions executed by those participants—the past remains secure. This is achieved by the use of ephemeral keys whose corresponding secrets are erased on successful completion of the exchange protocol. *Weak* PFS implies that this PFS is only achieved if adversaries cannot interfere with the protocol during the exchange (e.g., man-in-the-middle attacks), they can only attack it after the fact.

- Key compromise impersonation (KCI) resistance captures the scenario where an adversary reveals or corrupts the long-term secret key of a participant A: the adversary should be unable to impersonate other parties *to* A (but of course, can still impersonate A to other parties). For example, if Carol has compromised Alice's secret keys, she should be unable to send messages to Alice that Alice believes came from an uncorrupted third party, Bob.

- The maximal exposure (MEX) property states that, when given any one (long-term or ephemeral) secret key of each party in an exchange, the adversary should still be unable to distinguish the real session key from random. This property essentially takes into account all other combinations of keys that may be compromised in practice, hence the "maximal" denomination.

Standard security models generally define keys to be either long-term or ephemeral. As a recipient in the Signal protocol uses up to three keys, including a semi-static (medium-term) key, it is not at first obvious how to integrate this semi-static key into such two-key models. We choose to consider it as both long-term and ephemeral in different situations. This is discussed further in Remark 1.

We define the formal Key Indistinguishability Experiment now. We then provide a proof of security of our construction in this model in Section 6.

4.1 Key indistinguishability experiment

Let $\mathcal{K}$ denote the space of all possible session keys that could be derived in an exchange between two parties. We model n parties $P_1, \ldots, P_n$ through oracles Π_i^j, denoting the j-th session run by participant P_i. We limit the number of sessions per party by $1 \leq j \leq S$. Each oracle has access to the secret key of the corresponding party P_i's fixed long-term identity key IK_i, as well as the secrets for each of the m semi-static keys $\mathsf{SK}_i^1, \ldots, \mathsf{SK}_i^m$. Each oracle also has the following local variables:

- $\Pi_i^j.\mathsf{rand}$: The fixed randomness of oracle i for its j-th session (where Π_i^j is deterministic based on this randomness).
- $\Pi_i^j.\mathsf{role} \in \{\perp, \texttt{init}, \texttt{resp}\}$: The role of participant i in their j-th exchange.
- $\Pi_i^j.\mathsf{sk\_id}$: The index ℓ of the semi-static key SK_i^ℓ that participant i uses in their exchange j.
- $\Pi_i^j.\mathsf{peer\_id}$: The index k of the alleged peer P_k in the j-th exchange of oracle i.
- $\Pi_i^j.\mathsf{peer\_sk\_id}$: The index ℓ of the alleged peer's semi-static key $\mathsf{SK}_{\mathsf{peer\_id}}^\ell$ used in the exchange.
- $\Pi_i^j.\mathsf{sid}$: The session ID, explained further below.
- $\Pi_i^j.\mathsf{status} \in \{\perp, \texttt{accept}, \texttt{reject}\}$: Indicates whether the oracle has completed this session of the key exchange protocol and computed a session key from the exchange.
- $\Pi_i^j.\mathsf{session\_key} \in \mathcal{K}$: The computed session key.

These values are all initialised to $\perp$ at the start of the security experiment, except rand, which is initialised with random coins for each oracle. The oracle status is set to $\texttt{accept}$ or $\texttt{reject}$ on the computation of $\mathsf{session\_key}$.

The session ID is a feature of the security experiment, not the real protocol. We define the session ID to be a tuple $(\Pi, \mathsf{IK}_{\mathcal{I}}, \mathsf{IK}_{\mathcal{R}}, \mathsf{SK}_{\mathcal{R}}, \mathsf{EK}_{\mathcal{I}}, \boxed{\mathsf{EK}_{\mathcal{R}}})$ where $\mathcal{I}, \mathcal{R}$ denote the initiator and responder respectively, Π is a protocol identifier, and $\mathsf{EK}_{\mathcal{R}}$ is optional (so may be null). We say two sessions with the same sid are *matching*. This is done to restrict the adversary from making queries against any session matching the test session for the game—to avoid trivialising security. For a session Π_i^j we also define a *partner* session to be any session Π_k^ℓ for which $\Pi_i^j.\mathsf{peer\_id} = k$ and $\Pi_k^\ell.\mathsf{peer\_id} = i$, $\Pi_i^j.\mathsf{role} \neq \Pi_k^\ell.\mathsf{role}$, and $\Pi_i^j.\mathsf{sid} = \Pi_k^\ell.\mathsf{sid}$. We say any two such sessions are *partners*. Note that if two sessions are partners, they are also, by definition, matching.

Setup The security game is played between challenger $\mathcal{C}$ and a probabilistic polynomial-time (PPT) adversary $\mathcal{A}$. $\mathcal{C}$ will generate identity keys for the n participants, $\mathsf{IK}_1, \ldots, \mathsf{IK}_n$, and for each participant i, generate m semi-static keys $\mathsf{SK}_i^1, \ldots, \mathsf{SK}_i^m$. $\mathcal{C}$ will finally choose a uniformly random secret bit $b \leftarrow \{0, 1\}$, and provide $\mathcal{A}$ with access to the oracles Π_i^j.

Game Adversary $\mathcal{A}$ can adaptively make the following queries in the game:

- **Send**(i, j, μ): Send an arbitrary message μ to oracle Π_i^j. The oracle will behave according to the key exchange protocol and update its status appropriately.
- **RevealIK**(i): Return the secret long-term key of participant i. After this, participant i is *corrupted.*
- **RevealSK**(i, ℓ): Return the ℓ-th secret semi-static key of participant i. After this, SK_i^ℓ is said to be *revealed.*
- **RevealEK**(i, j): Return the ephemeral key (i.e., the random coins) of the j-th session of participant i. After this, EK_i^j and $\Pi_i^j.\mathsf{rand}$ are said to be *revealed.*
- **RevealSessionKey**(i, j): Return $\Pi_i^j.\mathsf{session\_key}$. After this, session Π_i^j is said to be *revealed.*

Test At some point in the game, $\mathcal{A}$ will issue a special $\textbf{Test}(i, j)$ query exactly once. $\mathcal{C}$ will return K_b to the adversary, where $K_0 := \Pi_i^j.\mathsf{session\_key}$ and $K_1 \leftarrow \mathcal{K}$ (a random key from the keyspace). After this query is made, session Π_i^j is said to be *tested.* $\mathcal{A}$ can continue to adaptively make queries to the above game functions after the Test query has been issued. Finally, $\mathcal{A}$ outputs a bit $b^* \in \{0, 1\}$ as their guess.

At this point, the tested session Π_i^j must be *fresh.* Freshness is defined in Definition 5, and the cases for freshness are also summarised in Table 2 for clarity.

Definition 5 (Freshness)**.** *A session* Π_i^j*, with* $\Pi_i^j.\mathsf{peer\_id} = k$*, is* ***fresh*** *if* none *of the following hold:*

- $\Pi_i^j.\mathsf{status} \neq \mathtt{accept}$*.*
- *The* session_key *of* Π_i^j*, or any matching session, is* revealed*.*
- *If* $\Pi_i^j.\mathsf{role} = \mathtt{init}$*:*
  - *Both* $\textbf{RevealIK}(i)$ *and* $\textbf{RevealEK}(i, j)$ *are issued.*
  - Π_i^j *has a partner* Π_k^ℓ *for some* ℓ*,* $\textbf{RevealIK}(k)$ *is issued, and either* $\textbf{RevealSK}(k, \Pi_i^j.\mathsf{peer\_sk\_id})$ $(\star)$ *or* $\textbf{RevealEK}(k, \ell)$ *are issued. See Remark 1.*
- *If* $\Pi_i^j.\mathsf{role} = \mathtt{resp}$*:*
  - Π_i^j *has a partner* Π_k^ℓ *for some* ℓ *and both* $\textbf{RevealIK}(k)$ *and* $\textbf{RevealEK}(k, \ell)$ *are issued.*
  - $\textbf{RevealIK}(i)$ *and either* $\textbf{RevealSK}(i, \Pi_i^j.\mathsf{sk\_id})$ $(\star)$ *or* $\textbf{RevealEK}(i, j)$ *are issued. See Remark 1.*
- Π_i^j *has no partner session and* $\textbf{RevealIK}(\Pi_i^j.\mathsf{peer\_id})$ *is issued.*

To define security in this model, we require correctness and soundness. Soundness ensures that, if the adversary is restricted to making only reveal queries that keep the test session **fresh**, then its advantage in distinguishing the session key from random is negligible. Let **fresh**(session) return true if session is fresh, and false otherwise.

Definition 6. *Let* $\mathcal{A}$ *be a PPT adversary. We define the advantage of* $\mathcal{A}$ *in winning the above key indistinguishability experiment* kie *with* n *parties,* m *semi-static keys per party, and* S *sessions per party, as*

$$\mathsf{Adv}^{\mathrm{kie}}_{n,m,S}(\mathcal{A}) = \left| \Pr\left[b = b^* \,\wedge\, \textbf{fresh}(\mathsf{test\_session}) \right] - \frac{1}{2} \right| .$$

An authenticated key exchange protocol Π *is secure in the Signal-adapted-CK model if it is:*

- ***Correct****: Any two parties following the protocol honestly derive the same* sid*,* session_key*, and both arrive at an* `accept` *state.*
- ***Sound****: The advantage of any PPT adversary* $\mathcal{A}$ *is* $\mathsf{Adv}^{\mathrm{kie}}_{n,m,S}(\mathcal{A}) \leq \mathsf{negl}$*.*

We emphasise that Table 2 and our definition of freshness in Definition 5 are strictly weaker than the standard eCK/CK+ cases and definitions—specifically, we have removed the adversary's ability to perform two specific cases of KCI attack. Both of these removed cases are given in Table 3, and correspond to the extra restrictions on freshness marked with a $(\star)$ in Definition 5. These are the cases that weaken the eCK/CK+ models to our Signal-adapted-CK model.

The reason for these exclusions from our model is that the original Signal X3DH protocol does not satisfy these properties, and our goal is to precisely model the security of that original protocol. Hence, these cases should be removed. The KCI attack on the original protocol is as follows: if Bob's semi-static key SK_B

is compromised, an adversary can impersonate anyone to Bob. This is because Alice is only authenticated through dh_1 (the exchange with SK_B), so an adversary can claim the use of any other public key IK_E and calculate the correct Diffie–Hellman value with SK_B. As SK_B is periodically replaced by Bob, the impersonation to Bob can last only as long as he accepts exchanges with that particular SK_B. However, we consider this a failure of the KCI property because SK_B is not ephemeral. This is discussed further in Remark 1.

Event	Matching session exists	$\mathsf{IK}_{\mathcal{I}}$	$\mathsf{EK}_{\mathcal{I}}$	$\mathsf{IK}_{\mathcal{R}}$	$\mathsf{SK}_{\mathcal{R}}$	$\mathsf{EK}_{\mathcal{R}}$	Attack
E_1	No	✓	x	x	✓	-	KCI
E_2	No	x	✓	x	x*	-	MEX
E_3	No	x	-	x	x*	✓	MEX
E_5	Yes	✓	x	✓	x	x	wPFS
E_6	Yes	x	✓	x	x*	✓	MEX
E_7	Yes	✓	x	x	✓	✓	KCI

Table 2: Behaviour of the adversary in our model, corresponding to the various freshness conditions in Definition 5. $\mathcal{I}$ and $\mathcal{R}$ denote whether the key belongs to the initiator or responder respectively. "✓" means the corresponding secret key is revealed or corrupted, "x" means it is not revealed, and "-" means it does not exist or is provided by the adversary.
*Discussed further in Remark 1.

Event	Matching session exists	$\mathsf{IK}_{\mathcal{I}}$	$\mathsf{EK}_{\mathcal{I}}$	$\mathsf{IK}_{\mathcal{R}}$	$\mathsf{SK}_{\mathcal{R}}$	$\mathsf{EK}_{\mathcal{R}}$	Attack
E_4	No	x	-	✓	✓	x	KCI
E_8	Yes	x	✓	✓	✓	x	KCI

Table 3: The two cases of the eCK/CK+ model which are NOT satisfied by Signal's X3DH, and so are not included in our model. This lack of KCI is exactly where these protocols break down.

Remark 1. In the original Signal X3DH protocol, the semi-static keys SK_B are used to strike a balance between perfect forward secrecy and key-exhaustion denial of service. To correctly model the purpose of this key, we assume it is "ephemeral enough" to have been replaced some time before a PFS attack (event E_5 in Table 2) takes place—this is generally a longer-term attack and the cycling of the semi-static key is designed to prevent this precise attack.

Because the semi-static key is reused and not actually ephemeral, we do not assume it is simply a long-term key in the other events of Table 2. In the KCI attacks, we allow it to be revealed as both ephemeral and long-term, to properly capture various forms of key-leakage that could lead to that attack and to strengthen the model (as mentioned above).

The MEX cases are more interesting, however. The original Signal X3DH protocol is not secure if the semi-static key can be revealed in cases E_2, E_3, and E_6. Hence, they are set to x in Table 2 due to our goal of accurately capturing the security of this original Signal protocol. In the spirit of the MEX property, the protocol would ideally be secure even when these three cases allowed SK to be revealed—there is no reason to treat the semi-static key as long-term in these cases. As we will show later, our new protocol (SI-X3DH) is secure even if these three cases marked by asterisks are changed to ✓.

4.2 Further security properties

We briefly discuss (full) perfect forward secrecy (PFS) as opposed to just weak PFS, which is proved in the model above. Krawczyk [Kra05] shows that any two-message key exchange protocol authenticated via public keys (without a secure shared state already established) cannot achieve true perfect forward secrecy. Despite this, it is claimed in [MP16b] that X3DH can be considered to have PFS, assuming that the identities of the users can be trusted via some means outside the protocol. In this specific case, Bob's signature on the semi-static key can be used to verify that the semi-static key does indeed belong to Bob, preventing even an active attacker from tampering with the keys Bob provides to defeat PFS (in particular, the server cannot maliciously provide semi-static keys to Alice while pretending they came from Bob). The same holds for our proposed scheme, but will not be discussed further in this thesis—the situation is identical to the original Signal X3DH.

Another very important property of X3DH, which isn't captured by the above security model (or in general by the eCK or CK+ models), is that of *deniability*. Deniability has two flavours: offline and online deniability. A protocol is offline-deniable if an adversary can gain no non-repudiable evidence of message authorship from a transcript even if the long-term keys involved are compromised. On the other hand, online deniability means that even by interacting with the target (or colluding with another user with whom the target interacts), the adversary cannot gain any such evidence. A protocol satisfying both offline and online deniability is known as strongly-deniable. Unfortunately, the Signal protocol fails to achieve online-deniability, as shown by Unger and Goldberg [UG18]—although this notion is very difficult to obtain and arguably less important that offline-deniability. The first formal proof that offline-deniability is indeed achieved by Signal was given by Vatandas et al. [VGIK20].

The proof of offline-deniability for Signal carries over to our protocol in an essentially identical manner, because of how similar the two protocols are. The proof reduces to the Knowledge of DH (KDH) assumption and its variants (K2DH and EKDH) which informally state that it should be infeasible for an adversary, given as input public keys for which the secret keys are unknown, to output DH values and other public keys they do not know the secret key to, yet still satisfy relationships of the form $\mathsf{dh}_i = \mathrm{DH}(K_1, K_2)$ (where K_1, K_2 are public keys). We will not formally define the assumptions here, but refer the reader to [VGIK20]. We give a brief, informal outline of this proof in Section 6.4.

5 Using SIDH for post-quantum X3DH

Suppose, first, that we naively drop in SIDH as a replacement for DH in Figure 1. In order to prevent adaptive attacks from either party, it suffices to require proof that certain public keys are honestly generated (for example, requiring proof that said member knows the corresponding private key). In the case of EK_A, this could easily be done through an FO-like transformation [HHK17], as was done in the KEM known as SIKE [CCH$^+$].

However, upon further examination we notice that Bob's semi-static public key poses an issue. As Bob may be offline at the time of exchange, and this key will be reused across multiple iterations of the protocol, he cannot reveal the secret key to Alice. Even if EK_A is proven to be honestly generated, this would allow a concrete attack here in the CK security model despite Galbraith's [Gal18, A.3] claim that using an ephemeral key in the exchange introduces enough randomness to prevent information about the long-term secret being leaked. Precisely, in CK-type models, the adversary can use a reveal query on the private key of EK_A to essentially remove the protection it provides, and then perform an adaptive attack using a malicious semi-static key. The best we can hope for then is that he also provides a non-interactive proof of honest generation of SK_B. Unfortunately, because the key SK_B is regularly rotated, such a proof would have to be regenerated and reverified every time, and these proofs are not (currently) efficient enough to make this an attractive course of action.

Instead, we opt to modify the original X3DH protocol somewhat, so that SK_B is not used in a key exchange with IK_A (temporarily removing dh_1 from Figure 1, which we shall soon replace). This means that even if Bob maliciously adapts SK_B in order to learn Alice's key, the only key he will learn is the secret to EK_A, which is ephemeral and revealed to him using the FO transform anyway. The other components, $\mathsf{dh}_2, \mathsf{dh}_3$, and dh_4, all involve only Alice's provably honest ephemeral key, so neither party can learn anything in these exchanges. Therefore, the only thing left to resolve is how to replace dh_1 so that IK_A is still used safely to implicitly authenticate Alice. We cannot use an exchange $\mathrm{SIDH}(\mathsf{IK}_A, \mathsf{EK}_B)$ for a symmetrical reason to above, even if we ignored the fact that EK_B is only optional. Thus, to include the key IK_A in the exchange to authenticate Alice, we are left only with one option: $\mathsf{dh}_1 = \mathrm{SIDH}(\mathsf{IK}_A, \mathsf{IK}_B)$.

In this case, we must prove that the long-term keys $\mathsf{IK}_A, \mathsf{IK}_B$ are honestly generated, to ensure an adaptive attack cannot be performed by registering multiple fake users with adaptive public identity keys. Because these keys are fixed and registered (or even authenticated) in advance, we do not encounter the efficiency degradation of using a more expensive proof to prove knowledge of the corresponding secret keys—a proof would have to be verified only once per new contact. In fact, depending on the trust model we use for the server, the verification of these proofs could be offloaded to the server at registration time and would have no impact on users. If we do not wish to place such trust in the server, it is simple to verify these proofs out-of-band at the time of first communication with any new contact. In fact, the Signal X3DH protocol already assumes that participants will authenticate each other's identity public key via some unspecified external channel, depending on the desired trust model [MP16b]. The Signal Private Messenger app presents "safety numbers" and QR codes that can be used to verify contacts in-person. Thus, the introduction of these proofs does not change the trust model of Signal. Proving SIDH public keys are honestly generated can be done using a non-interactive zero-knowledge (NIZK) Proof of Knowledge (PoK) of the corresponding secret key. De Feo, Dobson, Galbraith, and Zobernig [DDGZ21] present such a proof protocol and show that using it as part of a non-interactive key exchange is much more efficient than resorting to other protocols such as k-SIDH (in terms of isogeny computations) or generic NIZK proof systems. Thus, this SIDH PoK is perfectly suitable for our situation.

Exactly as in Signal's X3DH, we still also require a signature by Bob on SK_B, to ensure that the server does not fake SK_B and break perfect forward secrecy by later corrupting IK_B (one of the adversarial abilities in our security model). This poses another obstruction to efficiency, because using an SIDH signature here would require sending and verifying such a signature regularly—every time Bob replaces his semi-static key. SIDH signatures are inefficient, and we do not recommend their use for practical systems where signatures need to be regularly created and verified. Instead, we suggest using another post-quantum signature scheme, such as a hash-based signature. The ability to use any post-quantum signature scheme for this purpose was already discussed in Section 1.1. Whichever verification key Bob uses for these signatures should be registered (and verified) in advance, just as the identity keys are.

If IK_A and IK_B are proven to be honestly generated then we can use $\mathsf{dh}_1 = \mathrm{SIDH}(\mathsf{IK}_A, \mathsf{IK}_B)$ in the exchange without risk of adaptive attack. Historically, $H(E_{AB}, E_{XY})$ type protocols are referred to as the "unified model". A naive scheme of this form was shown to be vulnerable to interleaving and known-key attacks by Blake-Wilson, Johnson, and Menezes [BWJM97, Protocol 3]. Essentially, the adversary starts two sessions with the same user: $\Pi^s_{i,j}$ and $\Pi^u_{i,j}$ (participant i thinking they are communicating with j for the s- and u-th time, respectively). In each of these two sessions, the ephemeral key E_u (or E_s) provided by i is forwarded to the other session, and given back to i (as if coming from j). Then the shared key of both sessions will be $H(E_{ij}, E_{us})$. Revealing either of the two session keys will reveal the session key of the other. For comparison, a protocol of the form $H(E_{AY}, E_{BX})$ has that $H(E_{js}, E_{iu}) \neq H(E_{ju}, E_{is})$, so the attack would not be possible. Including the ephemeral keys E_s and E_u individually in the hash too would prevent this attack, because the ordering would differ between the two sessions. Jeong, Katz, and Lee [JKL04] prove this to be secure ($\mathcal{TS}2$) in the ROM provided knowledge of the secret keys is proven. In the Signal case, because we additionally have $\mathsf{dh}_2 = \mathrm{SIDH}(\mathsf{EK}_A, \mathsf{IK}_B)$ in the exchange, this symmetry between sender and receiver is already broken. Therefore, we claim that our modified dh_1 computation is secure.

One other disadvantage of this modification is that it impacts the KCI resistance of the scheme. That is, if the adversary corrupted IK_B, they could pretend to be Alice by choosing any ephemeral key they like, and calculating dh_1 using the known secret key, so Bob would accept it as coming from Alice herself. However, as above, this was the case with the original Signal X3DH anyway (if SK_B was corrupted). It is important to note that due to this modification, the impersonation can persist for longer than in X3DH, since corruption is no longer repaired by the regular replacement of SK_B. While worthy of consideration, we believe the change is acceptable. As mentioned in the introduction of this chapter, medium-term impersonation seems just as damaging as long-term, and corruption of an identity key is a severe break in security anyway. Because neither scheme can claim to have KCI resistance, we still assert that SI-X3DH satisfies the same security requirements as Signal X3DH, despite this practical difference.

Unlike traditional Diffie–Hellman, where both participants' keys are of the form g^x, in SIDH we have an asymmetric setup—one user uses a degree-$\ell_1^{e_1}$ isogeny, while the other uses a degree-$\ell_2^{e_2}$ isogeny. In order to make this work in X3DH where users can be both initiators and receivers, we require that each user has two long-term identity keys: one of each degree. For concreteness, we shall assume that $\ell_1 = 2$ and $\ell_2 = 3$, therefore the isogenies used by Alice and Bob have degree 2^{e_1} and degree 3^{e_2} respectively. The 3^{e_2}-isogeny key is used when initiating a key exchange (that is, by Alice), and the 2^{e_1}-isogeny key is used by the receiver (Bob), so that there is no ambiguity or incompatibility. This arrangement is chosen so that the sender has a slightly higher computational burden than the receiver.

All the semi-static keys Bob uploads to the third-party keyserver should thus be generated from 2^{e_1}-isogenies, as should his one-time (ephemeral) keys be. Whenever Alice initiates a key exchange, her ephemeral key should be a 3^{e_2}-isogeny key. Then all three (or four) SIDH exchanges used in the protocol will work as usual.

Thus, we arrive at our modified protocol, which we call **SI-X3DH** (Supersingular Isogeny X3DH). The protocol is given in Figure 3. In each instance of the protocol, Alice requests Bob's public key package from the server, as before. This key package includes Bob's signature verification key VK_B, which is used to validate the signature on his semi-static key SK_B. Alice will then generate a random seed s and use a preimage resistant hash function H_1 to compute an ephemeral secret key $\mathsf{sk}_e \leftarrow H_1(s)$. The corresponding public key is $\mathsf{EK}_A = E_0/\langle P_1 + [\mathsf{sk}_e]Q_1\rangle$ (where E_0, P_1, Q_1 are the base curve and $\ell_1^{e_1}$-torsion basis from the SIDH public parameters). She will then compute the pre-shared key PSK, and an FO-proof π as follows:

$$\begin{aligned}
&\mathsf{dh}_1 = \mathrm{SIDH}(\mathsf{IK}_A, \mathsf{IK}_B),\\
&\mathsf{dh}_2 = \mathrm{SIDH}(\mathsf{EK}_A, \mathsf{IK}_B),\\
&\mathsf{dh}_3 = \mathrm{SIDH}(\mathsf{EK}_A, \mathsf{SK}_B),\\
&\boxed{\mathsf{dh}_4 = \mathrm{SIDH}(\mathsf{EK}_A, \mathsf{EK}_B)},\\
&\mathsf{PSK} = \mathrm{KDF}(\mathsf{dh}_1 \parallel \mathsf{dh}_2 \parallel \mathsf{dh}_3 \boxed{\parallel \mathsf{dh}_4}),\\
&\pi = s \oplus H_2(\mathsf{dh}_1) \oplus H_2(\mathsf{dh}_2) \oplus H_2(\mathsf{dh}_3) \boxed{\oplus H_2(\mathsf{dh}_4)}.
\end{aligned} \tag{1}$$

H_1 and H_2 are the same PRGs used in Section 3.2. The reason π takes this form will be clear from the security proof we present in Section 6.

Alice then sends (EK_A, π) to Bob, along with an identifier for herself, and information about which of his ephemeral keys she used in the exchange (if any). Bob can check π is valid and honest by re-computing PSK' using IK_A and EK_A, computing s' from π by XORing with the values $H_2(\mathsf{dh}_j)$ (for $j = 1, 2, 3$, and if used, 4), then recomputing $\mathsf{sk}'_e \leftarrow H_1(s')$, and checking that the corresponding public key is equal to EK_A. He computes PSK as in Equation 1. If the verification of π succeeded, both Alice and Bob can compute the shared secret $K = \mathrm{KDF}(s \parallel \mathsf{EK}_A \parallel \mathsf{PSK})$. However, if verification failed, Bob should instead choose a random $r \leftarrow \{0,1\}^n$ and compute $K = \mathrm{KDF}(r \parallel \mathsf{EK}_A \parallel \mathsf{PSK})$. This way, his key will not match the one Alice derives with overwhelming probability, and the exchange fails, with Alice learning no information about the cause of failure (or about Bob's secret keys).

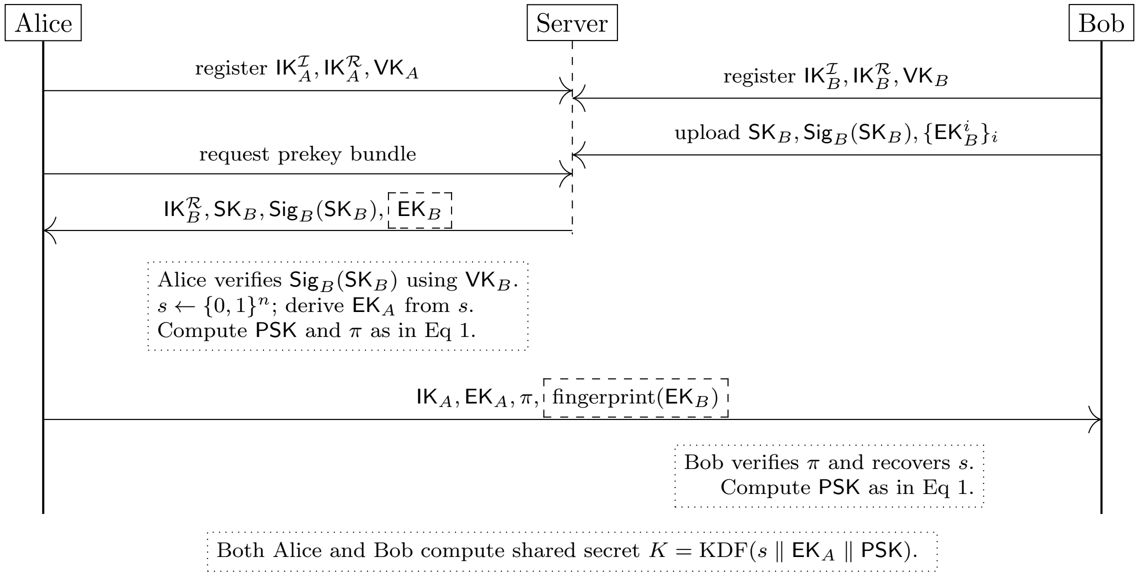

Alice
Server
Bob
register $\mathsf{IK}_A^{\mathcal{I}}, \mathsf{IK}_A^{\mathcal{R}}, \mathsf{VK}_A$
register $\mathsf{IK}_B^{\mathcal{I}}, \mathsf{IK}_B^{\mathcal{R}}, \mathsf{VK}_B$
upload $\mathsf{SK}_B, \mathsf{Sig}_B(\mathsf{SK}_B), \{\mathsf{EK}_B^i\}_i$
request prekey bundle
$\mathsf{IK}_B^{\mathcal{R}}, \mathsf{SK}_B, \mathsf{Sig}_B(\mathsf{SK}_B), \mathsf{EK}_B$
Alice verifies $\mathsf{Sig}_B(\mathsf{SK}_B)$ using VK_B.
$s \leftarrow \{0,1\}^n$; derive EK_A from s.
Compute PSK and π as in Eq 1.
$\mathsf{IK}_A, \mathsf{EK}_A, \pi, \mathsf{fingerprint}(\mathsf{EK}_B)$
Bob verifies π and recovers s.
Compute PSK as in Eq 1.
Both Alice and Bob compute shared secret $K = \mathrm{KDF}(s \parallel \mathsf{EK}_A \parallel \mathsf{PSK})$.

Figure 3: The SI-X3DH protocol.

6 Proof of security

Theorem 2. *The SI-X3DH protocol presented in Section 5 is secure (correct and sound) in the Signal-adapted-CK model of Definition 6, in the random oracle model (where H_1, H_2 and* KDF *are modelled as random oracles), assuming the SI-CDH problem is hard.*

Proof sketch We briefly outline the proof methodology. The proof is similar to the one given by Cohn-Gordon et al. [CGCD$^+$20], refitted to our Signal-adapted-CK model and using the Verifiable and Honest SI-CDH assumptions from Section 3.2 instead of the standard DDH oracle in the gap assumption. Cases E_2, E_3, and E_6 require IK_A and IK_B not to be revealed, so we use that as the basis for security in those cases. Similarly, cases E_1 and E_7 will use the fact that EK_A and IK_B are not revealed, and case E_5 relies on EK_A and SK_B not being revealed. Informally, the proof begins by forming a game in which the challenger guesses in advance which session will be tested, as well as the peer ID of that session. The challenger then simulates the game and inserts a VCDH or HCDH challenge into that predicted session, showing that an adversary winning the game can be used to successfully solve the respective hard problem. Once the cases are combined, this gives a proof of soundness of the SI-X3DH protocol.

Proof. It is clear that two parties following the protocol honestly will become partners. It is also clear that they will both successfully derive the same session key and enter an `accept` state, as an SIDH protocol has no failure probability if both parties are faithful. Thus the SI-X3DH protocol is *correct.*

To prove soundness, we will use a series of game hops. The proof will require splitting into cases following Table 2. Games 0 to 3 are common to all cases; we then break into a case-by-case proof.

Game 0. This game equals the security experiment in Section 4.1. The advantage of the adversary in this game is Adv_0. All queries to the random oracles (H_1, H_2, KDF) are simulated in an on-the-fly manner, and a table of (query, result) pairs is stored.

Game 1. We ensure all honestly generated SIDH keys are unique, or in other words, that there are no key collisions. If a key is generated that collides with any previously generated key, the challenger aborts and the adversary loses the game. With at most n parties, S sessions per party, m medium-term (semi-static) keys per party, we have at most $n + nm + nS$ receiving (2^{e_1}-isogeny) keys, and at most $n + nS$ sending (3^{e_2}-isogeny) keys. A collision among these keys is an instance of the generalised birthday problem, which we now briefly recall.

If M is the size of the domain from which $N \leq M$ objects are uniformly drawn, the generalised birthday problem shows that the probability of a collision between two objects is

$$p(N; M) = 1 - \prod_{k=1}^{N-1} \left(1 - \frac{k}{M}\right). \tag{2}$$

So,

$$\mathsf{Adv}_0 \leq p(n + nm + nS; |\mathcal{K}_2|) + p(n + nS; |\mathcal{K}_3|) + \mathsf{Adv}_1.$$

To be explicit, the size of an ℓ^e-isogeny keyspace is

$$(\ell + 1) \cdot \ell^{e-1}, \tag{3}$$

so $|\mathcal{K}_2| = 3 \cdot 2^{e_1 - 1}$ and $|\mathcal{K}_3| = 4 \cdot 3^{e_2 - 1}$. Note that the difference between Adv_0 and Adv_1 is therefore negligible, since the numerator in the collision probability is polynomially-sized while the denominator is exponential.

Game 2. We guess in advance which session Π_u^i the adversary will call the Test query against, and abort if this guess is incorrect. Note that we abort with high probability—there is only a $1/nS$ chance of success—but the advantages still only differ by a polynomial factor.

$$\mathsf{Adv}_1 = nS\mathsf{Adv}_2.$$

Game 3. In this game, we guess in advance the index of the peer of the test session Π_u^i—we guess a $v \in \{1, \ldots, n\}$ and abort if $\Pi_u^i.\mathsf{peer\_id} \neq v$. The probability of guessing v correctly is $1/n$, so

$$\mathsf{Adv}_2 \leq n\mathsf{Adv}_3.$$

We now split into cases based on Table 2. The cases will be grouped by the approach we take to reduce each case to the VCDH and HCDH hard problems. Specifically, in each scenario, we consider which of the SIDH exchanges is *not* compromised by reveal queries (that is, which of the edges in the exchange graph is not covered by the revealed vertices), and embed the hard problem into that pair of keys. Firstly, we address the MEX events, where neither IK_A nor IK_B are revealed—cases E_2, E_3, and E_6. We then treat the KCI events, cases E_1 and E_7, where EK_A and IK_B remain unrevealed. Finally, we come to the wPFS event, E_5, in which the adversary does not reveal either EK_A or SK_B.

We shall have, overall, that

$$\mathsf{Adv}_3 = \mathsf{Adv}_3^{2,3,6} + \mathsf{Adv}_3^{1,7} + \mathsf{Adv}_3^{5}.$$

6.1 Cases E_2, E_3, E_6 (MEX)

As mentioned above, the three cases E_2, E_3, and E_6 all rely on IK_A and IK_B not being revealed—the adversary should thus be unable to compute $\mathrm{SIDH}(\mathsf{IK}_A, \mathsf{IK}_B)$. This is the basis for the following part of the security proof.

Game 4. In this game, we abort if the adversary queries $\mathsf{dh}_1 = \mathrm{SIDH}(\mathsf{IK}_A, \mathsf{IK}_B)$ as the first component of a call to the KDF oracle. We call this event abort_4.

Whenever abort_4 occurs, we show that we can construct an algorithm $\mathcal{B}$ that can solve the Verifiable SI-CDH problem (VCDH) in Definition 3. As per that problem, $\mathcal{B}$ receives a triple $(E_A, E_B, \mathcal{O})$. $\mathcal{B}$ will simulate Game 3, except that it replaces IK_u with E_A and IK_v with E_B. It is guaranteed by freshness that $\mathcal{B}$ will never have to output the corresponding (unknown) secret keys. However, these two keys may be used in other sessions, so $\mathcal{B}$ must be able to behave in a consistent manner even when these keys are involved. Specifically, there are only two cases in which $\mathcal{B}$ is unable to compute the session key:

1. A non-tested session between the same users u, v where u is the initiator and v is the responder.
2. A non-tested session between any user other than u, and v, where v is the responder.

In the first of these two cases, the simulator does not know $\mathrm{SIDH}(E_A, E_B)$, which is needed for two reasons: $\mathcal{B}$ needs it to compute the session key, but it is also the solution to the VCDH challenge. In the second case, the simulator does not know $\mathrm{SIDH}(\mathsf{EK}_E, E_B)$ for potentially malicious ephemeral key EK_E, whose secret key is unknown to $\mathcal{B}$. In all other situations, $\mathcal{B}$ will know at least one of the secret keys involved in each SIDH exchange because they were all generated by the challenger.

We begin with the first case. If a session key or ephemeral key reveal query is made on such a session, $\mathcal{B}$ returns a random key. $\mathcal{B}$ also maintains a list of these random keys it generated, and correspondingly the public keys which *should* have been used to compute each one. Then, to ensure that other KDF queries made are consistent with these replaced keys, we do the following on receipt of a query $\mathrm{KDF}(\mathsf{dh}_1 \parallel \mathsf{dh}_2 \parallel \mathsf{dh}_3)$: $\mathcal{B}$ will query $\mathcal{O}(\mathsf{dh}_1)$, and if 1 is returned, this is exactly the case where abort_4 occurs—then $\mathcal{B}$ can return dh_1 as the answer to the VCDH challenge. Otherwise, $\mathcal{B}$ samples a new random key to return as the KDF response, and updates its list accordingly.

In the second case, we involve the FO-proof π_E also sent as part of the key exchange—a proof of honest generation for EK_E. In such a session, $\mathcal{B}$ will check through the output table of queries $\mathcal{A}$ has made to oracle H_2 (which can only have polynomially-many entries). Let IK_w be the identity key of the initiator. For each pair of entries (h, h'), we check whether $H_1(\pi_E \oplus h \oplus h' \oplus H_2(\mathrm{SIDH}(\mathsf{IK}_w, E_B)))$ is the secret key of EK_E. The simulator can always compute $\mathrm{SIDH}(\mathsf{IK}_w, E_B)$ when $w \neq u$ because it knows the private key for IK_w. In order for π_E to be valid, it must have the form

$$\pi_E = s_E \oplus H_2(\mathrm{SIDH}(\mathsf{IK}_w, E_B)) \oplus H_2(\mathsf{dh}_2) \oplus H_2(\mathsf{dh}_3)$$

so the only way for the adversary to have honestly generated π_E is for it to have queried H_2 on inputs $\mathsf{dh}_2, \mathsf{dh}_3$. Therefore, searching through all pairs (h, h') of queries will always result in recovery of s_E if π_E is valid, and if no such pair exists, the receiver would reject the FO-proof and fail the exchange. If such a pair is found, we can use the computed secret key to also compute $\mathrm{SIDH}(\mathsf{EK}_E, E_B)$. $\mathcal{B}$ can now use this j-invariant in a query to KDF to compute a consistent session key.

Thus, $\mathsf{Adv}(\mathsf{abort}_4) = \mathsf{Adv}^{\mathrm{vcdh}}(\mathcal{B})$ and

$$\mathsf{Adv}_3^{2,3,6} \leq \mathsf{Adv}^{\mathrm{vcdh}}(\mathcal{B}) + \mathsf{Adv}_4.$$

Game 5. In this game, we replace the session key of the test session with a uniformly random key. Because Game 4 aborts whenever a KDF oracle query is made involving dh_1, we know in this game that the adversary never queried KDF to get the true session key. Hence, the advantage of winning this game is

$$\mathsf{Adv}_4 = \mathsf{Adv}_5 = 0.$$

Therefore, we have

$$\mathsf{Adv}_3^{2,3,6} \leq \mathsf{Adv}^{\mathrm{vcdh}}(\mathcal{B}).$$

6.2 Cases E_1, E_7

These two cases rely on EK_A and IK_B not being revealed. Then $\mathsf{dh}_2 = \text{SIDH}(\mathsf{EK}_A, \mathsf{IK}_B)$ should be unknown to the adversary. The proof is very similar to the first cases above, but now relies on the Honest SI-CDH assumption from Definition 4. The main difference is that now, we must guess which of the signed semi-static keys will be used in the test session.

Game $4'$. In this game, the challenger guesses the index $j \in \{1, \dots, m\}$, such that signed semi-static key SK_v^j is used in the test session, and aborts if this guess is wrong. Consequently,

$$\mathsf{Adv}_3^{1,7} \leq m\mathsf{Adv}_{4'}.$$

Game $5'$ and $6'$. In Game $5'$, we abort if the adversary queries the KDF oracle with second component dh_2, equal to the test session's dh_2 component (derived from EK_u and IK_v). Once again, $\mathcal{B}$ will simulate Game $4'$. After receiving an HCDH instance triple (E_A, π, E_B), $\mathcal{B}$ will replace the ephemeral key of the test session with E_A, and IK_v with E_B. $\mathcal{B}$ will then also replace the test session FO-proof with $\pi_T := \pi \oplus H_2(\text{SIDH}(E_A, \mathsf{SK}_v^j)) \oplus H_2(\text{SIDH}(\mathsf{IK}_u, E_B))$. Recall from the definition of the HCDH problem, that π already includes the component $H_2(\text{SIDH}(E_A, E_B))$, as required, so π_T has the correct form.

There are two cases in which $\mathcal{B}$ will not be able to compute valid session keys for non-tested sessions. The first is for a session where any user initiates with $\mathsf{EK}_E \neq \mathsf{EK}_u$, and v is the responder. This is because $\text{SIDH}(\mathsf{EK}_E, E_B)$ is unknown when the secret key of EK_E is unknown. The second case is a special case of the first, when EK_u is reused in an exchange with v as the responder. As above, at least one secret key is known in all other situations, so these are the only two SIDH exchanges unable to be computed by $\mathcal{B}$.

In the first case, $\mathcal{B}$ will look up all pairs (h, h') in the polynomial-length output table of queries $\mathcal{A}$ has made to H_2. Suppose IK_w is the identity key of the initiator, and π_E is the FO-proof sent along with the ephemeral key EK_E. $\mathcal{B}$ will check whether $H_1(\pi_E \oplus h \oplus h' \oplus H_2(\text{SIDH}(\mathsf{IK}_w, E_B)))$ is the secret key of EK_E. As above, $\text{SIDH}(\mathsf{IK}_w, E_B)$ is known to $\mathcal{B}$ since the secret key of IK_w is. Also as above, the only way for the adversary to have generated a valid proof π_E is if they had made queries $H_2(\mathsf{dh}_2)$ and $H_2(\mathsf{dh}_3)$—otherwise, even if the adversary guessed the outputs of H_2 correctly (with negligible probability), they would not be able to verify that the π_E they created was actually correct without making the required queries to H_2 anyway. Hence, the only case the proof π_E is accepted is when a valid pair (h, h') exists in the query list of H_2, and if such a pair is found, we can use the secret key to compute the needed j-invariant $\text{SIDH}(\mathsf{EK}_E, E_B)$. $\mathcal{B}$ can now use this j-invariant in a query to KDF to compute a consistent session key. If no pair is found, the receiver would reject the FO-proof and fail the exchange.

In the second case, we cannot compute the output of KDF because $\mathsf{dh}_2 = \text{SIDH}(E_A, E_B)$ is unknown. So $\mathcal{B}$ will return a random key and keep a table for consistency as in the previous cases. Whenever the adversary makes a query to the KDF oracle, we check if $H_1(\pi \oplus H_2(\mathsf{dh}_2))$ corresponds to the secret key of E_A, and if it does, $\mathcal{B}$ has learned dh_2 as the SI-CDH value of E_A and E_B, this is also the case in which the game aborts. Note that the π used here is the one from the HCDH challenge, not from the exchange (π_E) or the test session (π_T). There is a negligible probability $1/2^n$ that the adversary guessed the correct output of H_2 without making a query of the form $H_2(\mathsf{dh}_2)$ (leading to an abort without recovering the answer to the HCDH challenge).

Game $6'$ is identical to Game 5 in the previous section. We therefore have

$$\mathsf{Adv}_3^{1,7} \leq m(\mathsf{Adv}^{\text{hcdh}}(\mathcal{B}) + 1/2^n).$$

6.3 Case E_5 (wPFS)

This case relies on EK_A and SK_B not being revealed (wPFS assumes that, in the future, these secrets are unrecoverable). Alternatively, this proof could be reduced to EK_A and EK_B which are both purely ephemeral.

However, because EK_B is optional in the Signal protocol (to avoid key exhaustion DoS), we reduce to the former scenario. In this case, we must again guess which of the signed semi-static keys will be used in the test session.

Game $4''$. In this game, the challenger guesses the index $j \in \{1, \ldots, m\}$, such that signed semi-static key SK_v^j is used in the test session. The game aborts if this guess is wrong. Hence,

$$\mathsf{Adv}_3^5 \leq n_m \mathsf{Adv}_{4''}.$$

Game $5''$ and $6''$. These proceed exactly as in Games $5'$ and $6'$ of cases E_1 and E_7 above, but with the HCDH challenge keys inserted into EK_u and SK_v^j. Furthermore, exactly as in the previous subsections, $\mathcal{B}$ knows the secret keys needed to compute the SIDH values of all exchanges except in two cases: an exchange with v as the responder using semi-static key SK_v^j (because EK_E is unknown and potentially maliciously chosen), and the specific subcase where $\mathsf{EK}_E = \mathsf{EK}_u$. This is essentially identical to cases E_1 and E_7. We conclude that

$$\mathsf{Adv}_3^5 \leq m(\mathsf{Adv}^{\text{hcdh}}(\mathcal{B}) + 1/2^n).$$

Finally, bringing all the game hops and cases together, we have

$$\begin{aligned}\mathsf{Adv}_{n,m,S}^{\text{kie}} \leq\ & p(n + nm + nS; |\mathcal{K}_2|) \\ & + p(n + nS; |\mathcal{K}_3|) \\ & + n^2 S\big[\mathsf{Adv}^{\text{vcdh}} + 2m\mathsf{Adv}^{\text{hcdh}} + m/2^{n-1}\big],\end{aligned} \tag{4}$$

where n is the number of participants, m is the number of semi-static keys per participant, and S is the maximum number of sessions run per party.

Because the VCDH and HCDH problems are hard if the SI-CDH problem is (shown in Section 3.2), it directly follows that SI-X3DH is secure if the standard SI-CDH problem is hard. □

6.4 Deniability

As mentioned in Section 4.2, the proof of offline-deniability of SI-X3DH is almost identical to that of the original Signal X3DH protocol (given in [VGIK20]), due to the similarity between the schemes. We just give a brief informal outline of the proof below.

Proof outline: Intuitively, for Bob to prove Alice's involvement, he would have to provide a Diffie–Hellman value $\mathrm{DH}(A, \cdot)$ which he could not possibly have generated himself—it must therefore have been generated by Alice. Because no DH values are exchanged between Alice and Bob in X3DH or SI-X3DH, and because the KDH, K2DH and/or EKDH assumptions hold, this is impossible. On top of this, because neither protocol uses a signature on session-specific information (unlike [HKKP21]), there is no loss of deniability there either. Proof of offline-deniability proceeds as an argument about simulatability, which we shall now sketch.

In the case of deniability for the initiator, given Alice's public key IK_A, the simulator Sim will generate $x \leftarrow \mathcal{K}_3$ and compute EK_A. Sim will then send this to Bob, who outputs keys $\mathsf{IK}_B, \mathsf{SK}_B, \mathsf{EK}_B$. The simulator can compute $\mathsf{dh}_2 = \mathrm{SIDH}(\mathsf{EK}_A, \mathsf{IK}_B)$, $\mathsf{dh}_3 = \mathrm{SIDH}(\mathsf{EK}_A, \mathsf{SK}_B)$, and $\mathsf{dh}_4 = \mathrm{SIDH}(\mathsf{EK}_A, \mathsf{EK}_B)$ because x is known, but cannot compute $\mathrm{SIDH}(\mathsf{IK}_A, \mathsf{IK}_B)$. Under the KDH-type assumptions, there must be an extractor $\hat{\mathcal{B}}$ for Bob's key IK_B—let us call it $\hat{\mathcal{B}}$. If $\hat{\mathcal{B}}$ outputs $\hat{Z}$ then the shared key is $\mathrm{KDF}(\hat{Z} \parallel \mathsf{dh}_2 \parallel \mathsf{dh}_3 \parallel \mathsf{dh}_4)$—the real shared key. On the other hand, if $\hat{\mathcal{B}}$ outputs $\perp$, then Sim chooses a session key at random. In either case, Sim also computes the FO-proof π using the session key it computed. In the second case, no PPT algorithm can compute SIDH($\mathsf{IK}_A, \mathsf{IK}_B$) without knowing IK_B, so the random key is indistinguishable from the real key.

We come now to the case of deniability for the responder, given Bob's public key IK_B, and also a signed semi-static key $\mathsf{SK}_B, \mathsf{Sig}_B(\mathsf{SK}_B)$. The simulator will send these two public keys to Alice, who outputs a key EK_A. Under the KDH-type assumptions, there exists an extractor $\hat{\mathcal{A}}$ for Alice which will either output the required SIDH values needed to compute the real key or will fail to output, in which case a random key will be indistinguishable from the real one as above. Thus, either way, assuming the KDH, K2DH and EKDH assumptions hold in the SIDH setting (which we claim they do), our SI-X3DH protocol is offline-deniable.

7 Efficiency

SIDH is a practically efficient post-quantum key exchange proposal. SIKE, derived from SIDH, is an alternate candidate in round 3 of NIST's post-quantum standardization competition. Duits [Dui19] examined the practical efficiency of using SIDH in the Signal protocol (though note that the implementation is not SI-X3DH, but the naive implementation, vulnerable to adaptive attacks), and found it entirely practical.

The SI-X3DH protocol uses three or four SIDH exchanges as part of the process to derive the shared key—a reflection of how Signal X3DH also uses three or four DH exchanges. In a single SI-X3DH exchange, the only other information sent (on top of the SIDH public keys) is the FO-proof π. This is simply n bits, which does not have a significant impact on the efficiency of the protocol. Thus, using SIDH for a post-quantum X3DH replacement is efficient at exchange time.

One of the main drawbacks of the SI-X3DH protocol is that it requires registering two keys rather than one on the server—a receiving key and a sending key. This is due to the inherent asymmetry of the SIDH protocol. However, SIDH has among the shortest key sizes of any post-quantum key exchange scheme, so this is not an issue. Note, too, that to initiate a conversation with a peer, only one key is required to be retrieved (the peer's sending key is not needed if they are the responder).

The second major drawback is that these keys also require an SIDH Proof of Knowledge or proof of honest generation, such as the one given by De Feo et al. [DDGZ21]. Depending on the trust model, this can be offloaded to the server at registration time or verified out-of-band, and only needs to be verified once. The best case is that a user verifies the proof for a contact once and then continues creating sessions with that same contact over a long period of time. However, if users regularly add new contacts, this could create a large overhead by requiring verification of such a proof for each. In the worse case, if a proof is required on nearly every new key exchange session, the overhead would be very large, and our scheme would no longer be efficient.

As discussed earlier, it appears that any post-quantum Signal X3DH replacement requires a post-quantum signature scheme to achieve perfect forward secrecy, and our scheme is no different. However, we emphasise that the use of a single signature is much more efficient than the generic schemes by Hashimoto et al. [HKKP21] and Brendel et al. [BFG$^+$22], which both require two signatures per exchange—one of which must be a more expensive ring or DVS signature to attain deniability.

We now consider the exchange-time efficiency of our protocol compared to the others proposed in the literature. By exchange-time, we mean the protocol occurring *after* the identity keys of the peer have been retrieved and verified (thus not taking into account the SIDH PoK on the identity keys). We consider the exchange-time efficiency because we assume a scenario in which we are beginning a new exchange with an already-verified peer, or a peer whose keys were verified in-person some time in advance.

As mentioned previously, our protocol is more efficient in terms of computation at exchange-time than Brendel et al.'s Split-KEM based X3DH [BFG$^+$20] protocol using CSIDH (assuming CSIDH does even satisfy the security properties needed for their split-KEM scheme, which they leave as an open problem). Based on NIST security level 1, we compare the fast, constant-time CTIDH [BBC$^+$21] implementation of CSIDH-512 with the SIKEp434 parameter set. According to Banegas et al. [BBC$^+$21], the cost of computing the CSIDH action is approximately 125 million Skylake clock cycles, while Cervantes et al. [COR21] state that SIKEp434 key generation and agreement takes around 5 million Skylake clock cycles—roughly 25 times faster. The split-KEM protocol proposed by Brendel et al. would require two CSIDH actions for each of the four encapsulations and decapsulations. SI-X3DH, on the other hand, requires only four SIDH exchanges, so in total would be around 50 times faster.

While the Signal-conforming AKE scheme proposed by Hashimoto et al. [HKKP21] and the SPQR protocol by Brendel et al. [BFG$^+$22] can be instantiated using efficient KEMs such as SIKE or other NIST post-quantum KEM candidates, the need for a post-quantum secure ring signature or DVS scheme is a large drawback to the efficiency of these protocols. Instantiating with the ring signature schemes of Beullens, Katsumata, and Pintore [BKP20], and choosing the lattice-based instantiation (Falafl) to optimise for speed (rather than signature and key size), would require around 78 million clock cycles for signing. Therefore, the signing time alone is already four times slower than the full SI-X3DH key exchange, and such a signature would be around 30 KB in size. The smaller isogeny-based instantiation (Calamari), whose signatures are around 3.6 KB, would take on the order of 10^{11} clock cycles—many orders of magnitude slower.

Thus, concretely, when performing an exchange with a user whose identity key has been verified via an SIDH Proof of Knowledge in advance or out-of-band, SI-X3DH is the fastest exchange-time post-quantum alternative to Signal's X3DH protocol currently in the literature.

Finally, to summarize the key differences with the original Signal X3DH protocol in a short form:

- Users must register two long-term public keys rather than one—a receiving and a sending key.
- Key compromise impersonation attacks (KCI) can no longer be rectified by replacing the semi-static key. Bob needs to switch to a new long-term key if his long-term key is compromised.
- Long-term key registration requires a proof of honest generation (such as [DDGZ21]), to avoid adaptive attacks by registering many fake users with malicious long-term keys.
- The signatures on Bob's semi-static keys can use any post-quantum signature scheme, and Bob should additionally register his signature verification public key so these can be validated.
- When initiating a new key exchange, Alice must also send a small FO-proof (n bits in size) along with her ephemeral public key, and Bob must check this proof on its receipt.

8 Conclusion

An SIDH key exchange is still safe for use if we have sufficient guarantee by both parties that their keys are honestly generated. This important observation allows us to use SIDH in a secure post-quantum replacement for Signal's X3DH protocol. We show that Brendel et al. [BFG$^+$20] were too rushed in dismissing SIDH as a candidate for this reason. While a naive drop-in use of SIDH into X3DH would be insecure as they claim, by tweaking the protocol to use a novel FO-like transform and a proof of knowledge for identity keys, we can make SIDH safe for use in the Signal X3DH protocol. Our new protocol, SI-X3DH, provides an efficient, post-quantum secure replacement for X3DH which closely resembles the original protocol.

References

[ACD19] Joël Alwen, Sandro Coretti, and Yevgeniy Dodis. The double ratchet: Security notions, proofs, and modularization for the Signal protocol. In *Advances in Cryptology – EUROCRYPT 2019*, pages 129–158, Cham, 2019. Springer International Publishing.

[AJL17] Reza Azarderakhsh, David Jao, and Christopher Leonardi. Post-quantum static–static key agreement using multiple protocol instances. In *International Conference on Selected Areas in Cryptography*, pages 45–63. Springer, 2017.

[BBC$^+$21] Gustavo Banegas, Daniel J. Bernstein, Fabio Campos, Tung Chou, Tanja Lange, Michael Meyer, Benjamin Smith, and Jana Sotáková. CTIDH: faster constant-time CSIDH. Cryptology ePrint Archive, Report 2021/633, 2021. `https://ia.cr/2021/633`.

[BFG$^+$20] Jacqueline Brendel, Marc Fischlin, Felix Günther, Christian Janson, and Douglas Stebila. Towards post-quantum security for Signal's X3DH handshake. In *Selected Areas in Cryptography–SAC 2020*, 2020.

[BFG$^+$22] Jacqueline Brendel, Rune Fiedler, Felix Günther, Christian Janson, and Douglas Stebila. Post-quantum asynchronous deniable key exchange and the Signal handshake. In *Public-Key Cryptography - PKC 2022 - 25th IACR International Conference on Practice and Theory of Public Key Cryptography, Virtual Event, March 8-11, 2022, Proceedings, Part II*, volume 13178 of *Lecture Notes in Computer Science*, pages 3–34. Springer, 2022.

[BKP20] Ward Beullens, Shuichi Katsumata, and Federico Pintore. Calamari and Falafl: Logarithmic (linkable) ring signatures from isogenies and lattices. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 464–492. Springer, 2020.

[BR93] Mihir Bellare and Phillip Rogaway. Entity authentication and key distribution. In *CRYPTO '93*, volume 773 of *Lecture Notes in Computer Science*, pages 232–249. Springer, Springer, 1993.

[BWJM97] Simon Blake-Wilson, Don Johnson, and Alfred Menezes. Key agreement protocols and their security analysis, 1997.

[CCH$^+$] Matthew Campagna, Craig Costello, Basil Hess, Amir Jalali, Brian Koziel, Brian LaMacchia, Patrick Longa, Michael Naehrig, Joost Renes, David Urbanik, et al. Supersingular isogeny key encapsulation.

[CGCD$^+$20] Katriel Cohn-Gordon, Cas Cremers, Benjamin Dowling, Luke Garratt, and Douglas Stebila. A formal security analysis of the Signal messaging protocol. *Journal of Cryptology*, 33(4):1914–1983, 2020.

[CK01] Ran Canetti and Hugo Krawczyk. Analysis of key-exchange protocols and their use for building secure channels. In *International conference on the theory and applications of cryptographic techniques*, pages 453–474. Springer, 2001.

[CLM$^+$18] Wouter Castryck, Tanja Lange, Chloe Martindale, Lorenz Panny, and Joost Renes. CSIDH: An efficient post-quantum commutative group action. In *Advances in Cryptology – ASIACRYPT 2018*, pages 395–427, Cham, 2018. Springer International Publishing.

[COR21] Daniel Cervantes-Vázquez, Eduardo Ochoa-Jiménez, and Francisco Rodríguez-Henríquez. Extended supersingular isogeny Diffie–Hellman key exchange protocol: Revenge of the SIDH. *IET Information Security*, 2021.

[Cre09] Cas J. F. Cremers. Formally and practically relating the CK, CK-HMQV, and eCK security models for authenticated key exchange. *IACR Cryptol. ePrint Arch.*, 2009:253, 2009.

[DDGZ21] Luca De Feo, Samuel Dobson, Steven D. Galbraith, and Lukas Zobernig. SIDH proof of knowledge. Cryptology ePrint Archive, Report 2021/1023, 2021. `https://ia.cr/2021/1023`.

[DFJP14] Luca De Feo, David Jao, and Jérôme Plût. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. *Journal of Mathematical Cryptology*, 8(3):209–247, 2014.

[DGL$^+$20] Samuel Dobson, Steven D. Galbraith, Jason LeGrow, Yan Bo Ti, and Lukas Zobernig. An adaptive attack on 2-SIDH. *International Journal of Computer Mathematics: Computer Systems Theory*, 5(4):282–299, 2020.

[Dui19] Ines Duits. The post-quantum Signal protocol: Secure chat in a quantum world. Master's thesis, University of Twente, 2019.

[FP21] Tako Boris Fouotsa and Christophe Petit. SHealS and HealS: Isogeny-based PKEs from a key validation method for SIDH. In *Advances in Cryptology - ASIACRYPT 2021 - 27th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 6-10, 2021, Proceedings, Part IV*, volume 13093 of *Lecture Notes in Computer Science*, pages 279–307. Springer, 2021.

[FSXY12] Atsushi Fujioka, Koutarou Suzuki, Keita Xagawa, and Kazuki Yoneyama. Strongly secure authenticated key exchange from factoring, codes, and lattices. In *Public Key Cryptography – PKC 2012*, pages 467–484, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[Gal18] Steven D. Galbraith. Authenticated key exchange for SIDH. Cryptology ePrint Archive, Report 2018/266, 2018. `https://eprint.iacr.org/2018/266`.

[GPST16] Steven D. Galbraith, Christophe Petit, Barak Shani, and Yan Bo Ti. On the security of supersingular isogeny cryptosystems. In *Advances in Cryptology – ASIACRYPT 2016*, pages 63–91. Springer Berlin Heidelberg, 2016.

[HHK17] Dennis Hofheinz, Kathrin Hövelmanns, and Eike Kiltz. A modular analysis of the Fujisaki–Okamoto transformation. In *Theory of Cryptography Conference*, pages 341–371. Springer, 2017.

[HKKP21] Keitaro Hashimoto, Shuichi Katsumata, Kris Kwiatkowski, and Thomas Prest. An efficient and generic construction for Signal's handshake (X3DH): Post-quantum, state leakage secure, and deniable. In *Public-Key Cryptography – PKC 2021*, pages 410–440, Cham, 2021. Springer International Publishing.

[JDF11] David Jao and Luca De Feo. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. In *Post-Quantum Cryptography*, pages 19–34, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[JKL04] Ik Rae Jeong, Jonathan Katz, and Dong Hoon Lee. One-round protocols for two-party authenticated key exchange. In *Applied Cryptography and Network Security*, pages 220–232, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg.

[KLM$^+$15] Daniel Kirkwood, Bradley C. Lackey, John McVey, Mark Motley, Jerome A. Solinas, and David Tuller. Failure is not an option: Standardization issues for post-quantum key agreement. Workshop on Cybersecurity in a Post-Quantum World, 2015.

[Kra05] Hugo Krawczyk. HMQV: A high-performance secure Diffie–Hellman protocol. In *Annual International Cryptology Conference*, pages 546–566. Springer, 2005.

[LLM07] Brian LaMacchia, Kristin Lauter, and Anton Mityagin. Stronger security of authenticated key exchange. In *International conference on provable security*, pages 1–16. Springer, 2007.

[MP16a] Moxie Marlinspike and Trevor Perrin. The double ratchet algorithm. `https://signal.org/docs/specifications/doubleratchet/`, 2016. Revision 1, 2016-11-20.

[MP16b] Moxie Marlinspike and Trevor Perrin. The X3DH key agreement protocol. `https://signal.org/docs/specifications/x3dh/`, 2016. Revision 1, 2016-11-04.

[Pei14] Chris Peikert. Lattice cryptography for the internet. In *Post-Quantum Cryptography*, pages 197–219, Cham, 2014. Springer International Publishing.

[Per16] Trevor Perrin. The XEdDSA and VXEdDSA signature schemes. `https://signal.org/docs/specifications/xeddsa/`, 2016. Revision 1, 2016-10-20.

[SSW20] Peter Schwabe, Douglas Stebila, and Thom Wiggers. Post-quantum TLS without handshake signatures. In *CCS '20: 2020 ACM SIGSAC Conference on Computer and Communications Security, Virtual Event, USA, November 9-13, 2020*, pages 1461–1480. ACM, 2020.

[UG18] Nik Unger and Ian Goldberg. Improved strongly deniable authenticated key exchanges for secure messaging. *Proceedings on Privacy Enhancing Technologies*, 2018(1):21–66, 2018.

[Vél71] Jacques Vélu. Isogénies entre courbes elliptiques. *C. R. Acad. Sci. Paris Sér. A-B*, 273:A238–A241, 1971.

[VGIK20] Nihal Vatandas, Rosario Gennaro, Bertrand Ithurburn, and Hugo Krawczyk. On the cryptographic deniability of the Signal protocol. In *Applied Cryptography and Network Security*, pages 188–209, Cham, 2020. Springer International Publishing.