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

September 15, 2021

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), 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 formally-defined security model or properties for the 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. The initial key agreement is currently done via a protocol known as Extended Triple Diffie-Hellman (X3DH) [MP16]. While Alwen et al. [ACD19] construct a version of the double ratchet component using KEMs that 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 Brendal et al. [BFG$^+$20] examines some of these challenges, and suggests that SIDH cannot be used to make X3DH post-quantum secure due to its vulnerability to adaptive attacks when static keys are used. For an isogeny-based variant of Signal they suggest using CSIDH [CLM$^+$18], however this primitive is much less efficient than SIDH in part due to subexponential quantum attacks that lead to much larger parameters.

Specifically, there is an adaptive attack on SIDH 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. This attack cannot be detected using tools such as pairings. One proposed method of overcoming this attack, known as k-SIDH [AJL17], runs many instances of the SIDH protocol in parallel and combines all the shared secrets using a hash function in the final step of the protocol. However, the adaptive 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. Clearly though, 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.

Other efforts to formalize the security properties of the Signal protocol and X3DH include the work by Cohn-Gordon et al. [CGCD$^+$20] on proving security of Signal, and the very recent work of Hashimoto et al. [HKKP21] proposing a generic security model (for what they call Signal-conforming AKEs) for the Signal initial key agreement and 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 also satisfy:

1. Correctness—if Alice and Bob complete an exchange together, they should derive the same key.
2. Secrecy / Key-indistinguishability under corruption of various combinations of participants' secret keys, for example the CK model [CK01] or the model in [CGCD$^+$20].
3. (Implicit) authentication of both parties.
4. Perfect forward secrecy (PFS).
5. It can be made asynchronous/non-interactive by hosting participants' public keys on a third party server, which is untrusted. The only possible malicious ability the server has is that it could deny Alice the retrieval of Bob's keys (or, say, not give out his one-time keys). This property is called *receiver obliviousness* in [HKKP21].
6. Identity-hiding / (Offline) deniability [VGIK20]—a transcript should not reveal the participants of the exchange.

Our first goal of this work is to show that SIDH can indeed be used to construct a post-quantum X3DH replacement, which satisfies the same security model as the original X3DH protocol, despite Brendal et al. [BFG$^+$20]'s claim. We propose a new, efficient, post-quantum key exchange protocol using SIDH, modelled after X3DH, which we call SI-X3DH. This new protocol solves the problem of adaptive attacks by using a variant of the Fujisaki-Okamoto (FO) transform to prove that Alice's ephemeral key is honestly generated, and a proof of well-formedness on each party's long-term keys, which only needs to be verified once (and could be offloaded to the PKI server depending on the trust model used). Because SIDH is an efficient post-quantum key exchange proposal with very small key sizes, and SI-X3DH requires only three or four SIDH exchanges (unlike k-SIDH), our protocol is also efficient and practical. For example, SIDH is much faster than CSIDH, used in Brendel et al. [BFG$^+$20]'s proposal, because CSIDH uses larger prime degree isogenies while SIDH commonly uses only 2 and 3 (including SIKE, although there are some variants which use other primes such as eSIDH [COR21]). Our scheme also does not rely on expensive machinery such as post-quantum ring signatures to achieve deniability (as [HKKP21] does). The efficiency of our scheme is discussed more in Section 7.

Due to the asymmetry between isogeny degrees in SIDH, our protocol requires users to register two keys rather than one (a receiving key and a sending key). SI-X3DH is agnostic to the signature scheme Signal uses to sign semi-static keys, so any efficient post-quantum signature scheme may be used alongside it—there is no restriction to use isogeny-based schemes. The only additional overhead required by SI-X3DH is 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). We also claim that SI-X3DH more closely models the original X3DH protocol in structure—for example, allowing Bob the balance between one-time keys (which may be exhausted leading to denial of service) and medium-term/semi-static keys which may provide less security in certain situations. These properties and differences are discussed further in Section 4.

While there are some differences between our protocol and X3DH (in particular, SI-X3DH uses DH(IK_A, IK_B) while previously X3DH used DH(IK_A, SK_B)—this notation is defined in Section 2), we give a proof of security for our new SI-X3DH protocol, and show that it satisfies the security properties of the original Signal protocol. In doing so we also discuss the main point of failure for X3DH not satisfying the eCK or CK+ security model—key-compromise impersonation (KCI) attacks—and the impact this change from DH(IK_A, SK_B) to DH(IK_A, IK_B) has on the security of the protocol.

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 do satisfy the security definitions of their split-KEM.

Very recently, Hashimoto et al. [HKKP21] presented their Signal-Conforming AKE construction, also using post-quantum KEMs to construct a generic Signal X3DH replacement.

We briefly outline the differences between these two works and this paper using the following table.

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	1 key, 3 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 Split-KEM protocol [BFG$^+$20] does not discuss the requirement for a signature scheme on the semi-static keys, but clearly 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 requires this signature for PFS too, for the same reason, but also uses a second signature from the key exchange initiator on the session ID—two signatures per exchange. 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.

For deniability, SC-AKE requires the initiator of the key exchange to sign the session ID. This signature creates non-repudiable evidence of the initiators involvement in the exchange. Hashimoto et al. suggest using a ring signature to attain deniability, however a post-quantum ring signature scheme is a much more expensive construction. Deniability of the split-KEM construction is not discussed by Brendel et al., and would appear to depend on how the split-KEM is instantiated.

Finally, it is important to note that the SC-AKE 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.

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

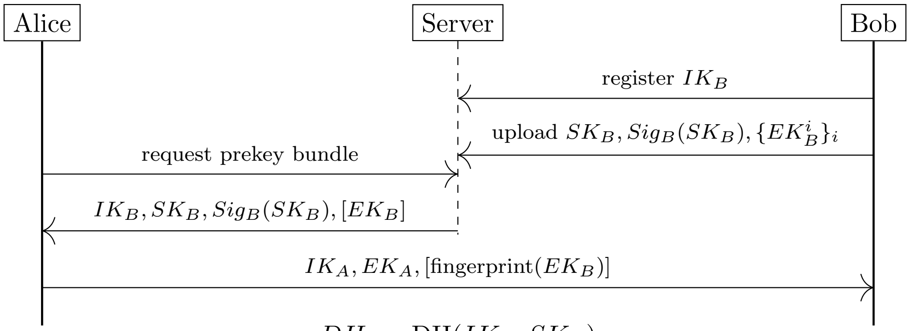
The basic process of the X3DH protocol is given in Figure 1, where Alice is the initiator and Bob is the responder. Let $\mathrm{DH}_g(g^a, g^b) = g^{ab}$ denote the result of a Diffie-Hellman key exchange between keys A and B (at least one of the private keys is needed to compute this, but the result is unambiguous).

Note that throughout this paper, we will use brackets $[X]$ to denote an optional parameter which may be omitted.

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, and are combined with any secure key derivation function (KDF), for example a secure 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


Alice
Server
Bob
register IK_B
upload $SK_B, Sig_B(SK_B), \{EK_B^i\}_i$
request prekey bundle
$IK_B, SK_B, Sig_B(SK_B), [EK_B]$
IK_A, EK_A, [fingerprint(EK_B)]

$$
\begin{gathered}
DH_1 = \mathrm{DH}(IK_A, SK_B) \\
DH_2 = \mathrm{DH}(EK_A, IK_B) \\
DH_3 = \mathrm{DH}(EK_A, SK_B) \\
[DH_4 = \mathrm{DH}(EK_A, EK_B)] \\
SK = \mathrm{KDF}(DH_1 \parallel DH_2 \parallel DH_3 \parallel [DH_4])
\end{gathered}
$$

Figure 1: The X3DH protocol [MP16]. DH_4 is optional on the basis of one-time key availability.

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 $p = \ell_1^{e_1} \cdot \ell_2^{e_2} \cdot f \pm 1$, where ℓ_1, ℓ_2 are 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 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 key exchange.

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.

[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.

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

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

Throughout this paper, we will use the function $\mathrm{SIDH}_{pp}(\cdot, \cdot)$ 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 $\mathrm{SIDH}_{pp}((E_A, P'_B, Q'_B), K_B) = j(E_A/\langle P'_B + [\beta]Q'_B \rangle)$.

3.1 SIDH assumptions

The standard computational and decisional hardness assumptions associated with the SIDH key exchange are as follows. Let

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

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

$$\mathrm{SSJ}_p = \{j(E_i) \,:\, E_i \text{ supersingular}\}$$

be the set of all possible supersingular j-invariants which could arise as shared secrets of an SIDH key exchange.

Let pp denote the public parameters $pp = (p, \ell_1, \ell_2, e_1, e_2, E_0, P_1, Q_1, P_2, Q_2)$.

Computational Supersingular-Isogeny Diffie-Hellman (SI-CDH) Problem. Given the public parameters pp, and two public keys $K_1 = (E_1, P'_1, Q'_1) \in \mathrm{SSEC}_{p,1}$, $K_2 = (E_2, P'_2, Q'_2) \in \mathrm{SSEC}_{p,2}$, compute the j-invariant $j = j(E_{12}) = j(E_{21}) = \mathrm{SIDH}_{pp}(K_1, K_2)$.

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

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

The SI-CDH assumption states that for any PPT adversary $\mathcal{A}$, $\mathsf{Adv}^{\text{si-cdh}}(\mathcal{A}) \leq \mathsf{negl}$

Decisional Supersingular-Isogeny Diffie-Hellman (SI-DDH) Problem. Let $K_1 = (E_1, P'_1, Q'_1) \in \mathrm{SSEC}_{p,1}$, $K_2 = (E_2, P'_2, Q'_2) \in \mathrm{SSEC}_{p,2}$ be two public keys, and j-invariant $j_0 = j(E_{12}) = j(E_{21}) = \mathrm{SIDH}_{pp}(K_1, K_2)$ be their SIDH shared secret. Let $j_1 \leftarrow_R \mathrm{SSJ}_p$. The SI-DDH problem is to distinguish between j_0, j_1, given K_1, K_2.

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

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

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

Gap assumption. It has been observed, for example by Galbraith and Vercauteren [GV18], that an oracle solving the SI-DDH problem can be used to solve the SI-CDH problem, making these two problems equivalent. This means there exists no typical Gap-DH problem in the SIDH setting. We discuss this issue and our workaround in Section 4.3.

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. 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 and 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 or most desirable models in this vein, 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 where 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, the result of at least one DH exchange is not able to be computed (because the adversary does not have either of the secret keys involved), so 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$^+$20], as there do not exist edges in the exchange graph for every pair of keys (e.g. there is no DH exchange between Alice's identity key and Bob's identity key or ephemeral key). Our benchmark for security is that a replacement protocol should meet at least the same security definition as that of the original protocol so we must observe where exactly the original protocol breaks down in the eCK model. This allows us to propose a slightly weaker model, though still stronger than the CK model, that successfully represents the security goals. This gives a more formal and well-defined security model than Cohn-Gordon et al. [CGCD$^+$20] used to prove security of the original Signal X3DH protocol, which used non-standard notation in an ad-hoc model. 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.

Before we begin, let us briefly recall 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 successfully 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. person-in-the-middle attack), they can only attack it after-the-fact.
- Key Compromise Impersonation (KCI) resistance captures the scenario where an adversary reveals/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).
- 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.

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

We model n parties $P_1, \ldots, P_n$ through oracles Π_i^j, which denotes 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 secret keys of the corresponding party P_i's fixed long-term identity key IK_i, as well as the m semi-static keys $SK_i^1, \ldots, SK_i^m$. Let $\mathcal{K}$ denote the space of session keys. Each oracle also has the following local variables:

- $\Pi_i^j.rand$—the fixed randomness of oracle i (which is deterministic based on this randomness) for its j-th session.
- $\Pi_i^j.role \in \{\bot, \texttt{init}, \texttt{resp}\}$—the role of participant i in their j-th exchange.
- $\Pi_i^j.sk\_id$—the index ℓ of the semi-static key SK_i^ℓ participant i uses in their exchange j.
- $\Pi_i^j.peer\_id$—the index k of the alleged peer P_k in the j-th exchange of oracle i.
- $\Pi_i^j.peer\_sk\_id$—the index ℓ of the alleged peer's semi-static key SK_k^ℓ used in the exchange.
- $\Pi_i^j.sid$—the session ID.
- $\Pi_i^j.status \in \{\bot, \texttt{accept}, \texttt{reject}\}$—indicates whether the oracle has completed the key exchange protocol and computed a session key for the exchange.
- $\Pi_i^j.session\_key \in \mathcal{K}$—the computed session key.

These values are all initialized to $\bot$ at the start of the security experiment, except *rand*, which is initialized with random coins for each oracle. The oracle status is set to `accept` or `reject` on computation of *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 $(\Pi, IK_\mathcal{I}, IK_\mathcal{R}, SK_\mathcal{R}, EK_\mathcal{I}, [EK_\mathcal{R}])$ where $\mathcal{I}, \mathcal{R}$ denote the initiator and responder respectively, Π is a protocol identifier, and $[EK_\mathcal{R}]$ is optional in the protocol 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 trivializing security. For a session Π_i^j we also define a *partner* session to be any session Π_k^ℓ for which $\Pi_i^j.peer\_id = k$ and $\Pi_k^\ell.peer\_id = i$, $\Pi_i^j.role \neq \Pi_k^\ell.role$, and $\Pi_i^j.sid = \Pi_k^\ell.sid$. 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 PPT adversary $\mathcal{A}$. $\mathcal{C}$ will generate identity keys for the n participants, $IK_1, \ldots, IK_n$, and for each participant i, generate m semi-static keys $SK_i^1, \ldots, SK_i^m$. $\mathcal{C}$ will finally choose a uniformly random secret bit $b \leftarrow \{0, 1\}$, and provide access to the oracles Π_i^j to $\mathcal{A}$.

Game $\mathcal{A}$ 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 accordingly.
- **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 $\Pi_i^j.rand$ are said to be *revealed.*
- **RevealSessionKey**(i, j)—return $\Pi_i^j.session\_key$. After this, session Π_i^j is said to be *revealed.*

Test At some point in the game, $\mathcal{A}$ will issue a special **Test**(i, j) query exactly once. $\mathcal{C}$ will return K_b to the adversary, where $K_0 := \Pi_i^j.session\_key$ 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 1, and the cases for freshness are also summarized in Table 2 for clarity.

Definition 1 (Freshness)**.** *A session Π_i^j, with $\Pi_i^j.peer\_id = k$, is **fresh** if* none *of the following hold:*

- $\Pi_i^j.status \neq \texttt{accept}$.
- *The session_key of Π_i^j, or any matching session, is revealed.*
- *If $\Pi_i^j.role = \texttt{init}$:*
  - *Both **RevealIK**(i) and **RevealEK**(i, j) are issued.*
  - *Π_i^j has a partner Π_k^ℓ for some ℓ, and **RevealIK**(k), and either **RevealSK**$(k, \Pi_i^j.peer\_sk\_id)$ $(\star)$ or **RevealEK**(k, ℓ) are issued. See Remark 1.*
- *If $\Pi_i^j.role = \texttt{resp}$:*
  - *Π_i^j has a partner Π_k^ℓ for some ℓ and both **RevealIK**(k) and **RevealEK**(k, ℓ) are issued.*
  - ***RevealIK**(i) and either **RevealSK**$(i, \Pi_i^j.sk\_id)$ $(\star)$ or **RevealEK**(i, j) are issued. See Remark 1.*
- *Π_i^j has no partner session and **RevealIK**(k) 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 which 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 2. *Let $\mathcal{A}$ be a PPT adversary. We define the advantage of $\mathcal{A}$ in winning the above key indistinguishability experiment* exp *with n parties, m semi-static keys per party, and S sessions per party, as:*

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

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 $\mathsf{Adv}^{\text{exp}}_{n,m,S}(\mathcal{A}) \leq \mathsf{negl}$.*

We emphasize that Table 2 and our definition of freshness in Definition 1 are strictly weaker than the standard eCK/CK+ cases/definitions—specifically, we have removed the adversary's ability to perform two specific cases of a KCI attack. Both these removed cases are given in Table 3, and correspond to the extra restrictions on freshness marked with a ($\star$) in Definition 1. These are the cases which weaken the eCK/CK+ models to our Signal-adapted-CK model.

This is because 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 excluded. 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 ID_E and calculate the correct Diffie-Hellman value with SK_B. Because 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 as SK_B is not ephemeral.

Event	Case	Matching session exists	IK_I	EK_I	IK_R	SK_R	EK_R	Attack
E_1	1	No	✓	x	x	✓	-	KCI
E_2	2	No	x	✓	x	x*	-	MEX
E_3	2	No	x	-	x	x*	✓	MEX
E_5	4	Yes	✓	x	✓	x	x	wPFS
E_6	5	Yes	x	✓	x	x*	✓	MEX
E_7	3	Yes	✓	x	x	✓	✓	KCI

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

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

Event	Case	Matching session exists	IK_I	EK_I	IK_R	SK_R	EK_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.

Because the semi-static key is reused and not actually ephemeral, though, 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, though, 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) PFS as opposed to just weak PFS, which is proved in the model above. Krawczyk [Kra05] shows that any 2-message key-exchange protocol authenticated via public keys (without a secure shared state already established) cannot achieve true Perfect Forward Secrecy (PFS). Despite this, it is claimed in [MP16] 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 paper—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 is essentially identical for our protocol, 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 $DH_i = \mathrm{DH}(P_i, Q_i)$ type relationships. 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.

4.3 New CDH-based assumptions

Let $H_1 : \{0,1\}^* \to \mathcal{K}_3$ be a PRF whose codomain is the 3-isogeny secret key space. We also let $H_2 : \{0,1\}^* \to \{0,1\}^\lambda$ be a PRF. Both H_1 and H_2 are modelled as random oracles. The function $\texttt{pubkey\_from\_secret}(\alpha)$ returns the public key corresponding to secret key α.

Verifiable CDH problem (VCDH) We define a slightly different assumption, similar to the Computational Diffie-Hellman (CDH) assumption, except with an additional "check" oracle provided by the challenge generator.

An instance of this problem is a triple $(EK_A, EK_B, \mathcal{O})$ where $EK_i \in \mathrm{SSEC}_{p,i}$ are public keys and $\mathcal{O}$ is a truth oracle defined as $\mathcal{O}(j') = (j' \overset{?}{=} \mathrm{SIDH}_{pp}(EK_A, EK_B)$.

Definition 3 (Verifiable CDH problem)**.** *Given a triple* $(EK_A, EK_B, \mathcal{O})$ *as above, output* $\mathrm{SIDH}_{pp}(EK_A, EK_B)$*.*

Essentially oracle $\mathcal{O}$ is an obfuscated point function, confirming if the answer to the CDH challenge is correct or not. So intuitively we should learn no extra information from this oracle—on all except one j-invariant the oracle will return false, so in polynomially-many queries, the likelihood of guessing the correct j-invariant is negligible (as in the CDH problem). We can prove a reduction to the SI-CDH problem in the random oracle model. Let $H(\cdot)$ be the point function obfuscator, where H is a random oracle. Then $\mathcal{O}(j')$ checks $H(j') = h$ where, $h = H(j)$. So if there is an adversary that makes q queries and wins, simply return one of the inputs queried to H, winning the SI-CDH game with $1/q$ probability.

Honest CDH problem (HCDH) This problem models a CDH instance with an additional FO-like proof that the first key (EK_A) was honestly generated.

An instance of this problem is a triple (EK_A, EK_B, π) where s is a uniformly random value in $\{0,1\}^\lambda$, and:

$$\pi = s \oplus H_2(\mathrm{SIDH}_{pp}(EK_A, EK_B))$$
$$EK_A = \texttt{pubkey\_from\_secret}(H_1(s)).$$

Instances of this form can be generated as in Algorithm 1, for example.

Algorithm 1: `Honest CDH (HCDH) challenge generator`

```
Input: Public parameters $pp$
Output: SIDH public keys $EK_A, EK_B$, FO-proof $\pi$ for $EK_A$
$s \leftarrow_R \{0,1\}^\lambda$
$\alpha \leftarrow H_1(s)$
$EK_A = \texttt{pubkey\_from\_secret}(\alpha)$
$\beta \leftarrow_R \mathcal{K}_2$
$EK_B = \texttt{pubkey\_from\_secret}(\beta)$
$PSK = \mathrm{SIDH}_{pp}(EK_A, EK_B)$
$\pi = s \oplus H_2(PSK)$
return $(EK_A, EK_B, \pi)$
```

Definition 4 (Honest CDH problem)**.** *Given a triple* (EK_A, EK_B, π) *as above, output* $\mathrm{SIDH}_{pp}(EK_A, EK_B)$*.*

We argue that the FO-like proof leaks no information because we obviously assume that $\mathrm{SIDH}_{pp}(EK_A, EK_B)$ is unknown (that is the answer to the CDH problem) and s is random, thus if the CDH problem is hard then so too is this problem. Again we give a reduction in the random oracle model. Treat H_1 and H_2 as random oracles, and choose π as a random binary string. Again one of the q queries to H_2 must be the correct j-invariant. So we just choose a random $1 \leq i \leq q$ and hope the i-th query to H_2 is the correct one, and stop the interaction on this query, returning the corresponding input to H_2. Again, this wins with probability $1/q$.

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—in this model the adversary can use a reveal query on the private key of EK_A to essentially remove the protection it provides. The best we can hope for then is that he also provides a non-interactive proof of honest generation of SK_B, for example a signature from SK_B which is a PoK, however this is undesirable due to the inefficiency of SIDH-type signatures.

Instead, we opt to modify the original X3DH protocol somewhat, so that SK_B is not used in a key exchange with IK_A (removing DH_1). This means that even if Bob adapts SK_B to learn Alice's key, the only key he will learn is EK_A which is ephemeral and revealed to him using FO anyway. DH_2, DH_3, and DH_4 all involve only Alice's provably honest ephemeral key so neither party can learn anything in these exchanges. So the only thing left to resolve is in how to replace DH_1 so that IK_A is still used safely to implicitly authenticate Alice. We cannot use an exchange SIDH(IK_A, EK_B) for a similar reason (even ignoring that EK_B is only optional). Thus, to include the key IK_A in the exchange to authenticate Alice, we are left only with the option $DH_1 = \text{SIDH}(IK_A, IK_B)$.

In this case, we must prove that the long-term keys IK_A, 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/authenticated in advance, we do not encounter the efficiency degradation of using a more expensive proof of each of these keys to prove knowledge of the secret key—this 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 a new contact. In fact, the Signal X3DH protocol already assumes that participants will authenticate each others public keys via some unspecified external channel, depending on the desired trust model [MP16]. Thus, these proofs do not change the trust model of Signal at all. 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 other protocols such as k-SIDH (in terms of isogeny computations) or generic NIZKs. Thus, their proof 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 doesn't fake SK_B and break weak perfect forward secrecy by later corrupting IK_B (one of the adversarial abilities in our security model). This poses another issue 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. Instead we suggest using another post-quantum signature scheme, such as a hash-based signature. Whichever key Bob uses to sign his pre-shared keys should just be registered in advance as the identity keys are.

If IK_A and IK_B are honestly generated then we can use $DH_1 = \text{SIDH}(IK_A, IK_B)$ in the exchange without risk of adaptive attack. Historically, $H(E_{AB}, E_{XY})$ type protocols are referred to as the "unified model." This naive scheme was shown to be vulnerable to interleaving and known key attacks in Protocol 3 of [BWJM97]. Essentially, the adversary starts two sessions from the same user, $\Pi^s_{i,j}$ and $\Pi^u_{i,j}$ (participant i thinking it is communicating with j for the s, u-th time). The ephemeral keys E_u and E_s are both forwarded to the other session. Then the shared key of both sessions will be $H(E_{ij}, E_{us})$. Revealing either will reveal the session key of the other. Compare this to the $H(E_{AY}, E_{BX})$ scenario where $H(E_{js}, E_{iu}) \neq H(E_{ju}, E_{is})$. 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. [JKL04] proves 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 $DH_2 = \text{SIDH}(EK_A, IK_B)$ in the exchange, then this symmetry is already broken. So we claim that our modified DH_1 computation is secure. One other disadvantage of this modification is that it does not provide KCI security. 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)—so while the impersonation can persist for longer than in X3DH (it isn't prevented by the regular replacement of SK_B), we believe that this change is not a major degradation in security.

Unlike traditional Diffie-Hellman, where both participants' keys are of the form g^x, in SIDH we have a more asymmetric setup—one user must use a degree 2^n-isogeny while the other uses a 3^m-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. The 3-isogeny key is used when initiating a key exchange (that is, by Alice), and the 2-isogeny key is used by the responder/receiver (Bob), so that there is no ambiguity or incompatibility. This is chosen so that the sender has a slightly higher computational burden than the receiver.

All of Bob's semi-static keys uploaded to the server should thus be generated from 2-isogenies, as should his one-time/ephemeral keys. Whenever Alice initiates a key exchange, her ephemeral key should be a 3-isogeny key. Then all three (or optionally four) SIDH exchanges will work.

Thus, we arrive at our modified protocol, which we call SI-X3DH (supersingular-isogeny X3DH). The protocol is given in Figure 3. As above, brackets in the protocol $[X]$ denote optional values (this cannot be confused with scalar multiplication of elliptic curve points from the context). For each instance of the protocol, Alice would request Bob's public key package from the Server as before. She will then generate a random seed s and use a pre-image resistant hash function H_1 to compute an ephemeral secret key $esk = H_1(s)$. The corresponding public key is $EK_A = E/\langle P_A + [esk]Q_A\rangle$ (where E, P_A, Q_A are the SIDH public parameters). She will then compute pre-shared key PSK and proof π as:

$$\begin{aligned}
DH_1 &= \text{SIDH}(IK_A, IK_B) \\
DH_2 &= \text{SIDH}(EK_A, IK_B) \\
DH_3 &= \text{SIDH}(EK_A, SK_B) \\
[DH_4 &= \text{SIDH}(EK_A, EK_B)] \\
PSK &= \text{KDF}(DH_1 \parallel DH_2 \parallel DH_3 \parallel [DH_4]) \\
\pi &= s \oplus H_2(DH_2) \oplus H_2(DH_3)\, [\oplus H_2(DH_4)\,]
\end{aligned} \tag{1}$$

She then sends (EK_A, π) to Bob, along with an identifier for herself and 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 $H_2(DH_{2,3,4})$, and then recomputing $esk' = H_1(s')$ and checking 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 $SK = \text{KDF}(s \parallel EK_A \parallel$

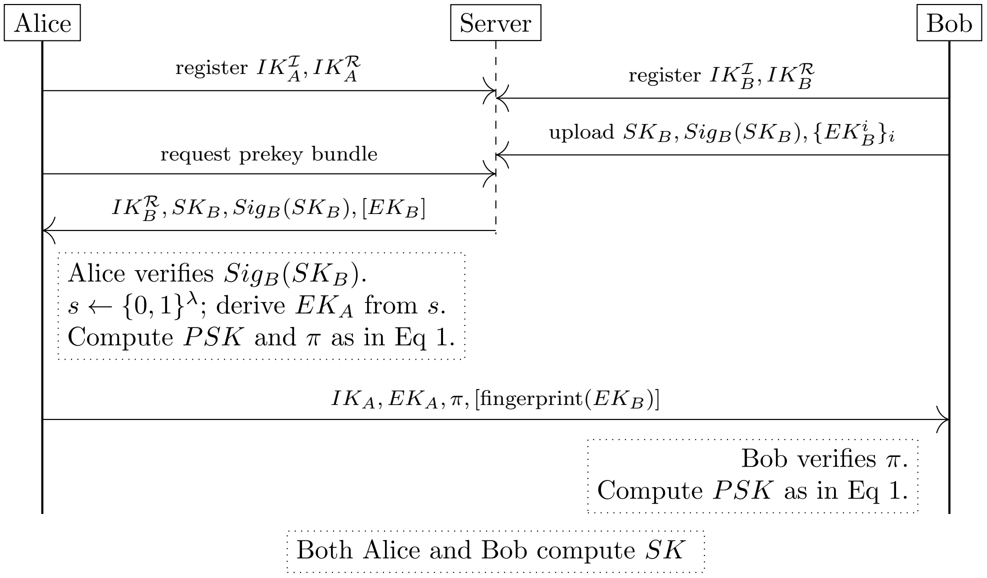

Alice
Server
Bob
register $IK_A^{\mathcal{I}}, IK_A^{\mathcal{R}}$
register $IK_B^{\mathcal{I}}, IK_B^{\mathcal{R}}$
upload $SK_B, Sig_B(SK_B), \{EK_B^i\}_i$
request prekey bundle
$IK_B^{\mathcal{R}}, SK_B, Sig_B(SK_B), [EK_B]$
Alice verifies $Sig_B(SK_B)$.
$s \leftarrow \{0,1\}^\lambda$; derive EK_A from s.
Compute PSK and π as in Eq 1.
IK_A, EK_A, π, [fingerprint(EK_B)]
Bob verifies π.
Compute PSK as in Eq 1.
Both Alice and Bob compute SK

Figure 3: The SI-X3DH protocol.

PSK). However, if verification failed, Bob should instead choose a random $r \leftarrow \{0,1\}^\lambda$ and compute $SK = \mathrm{KDF}(r \parallel EK_A \parallel PSK)$. This way, his key will not match Alice's and the exchange fails, while Alice learns no information about the cause of failure (or about Bob's secret keys).

6 Proof of security

Theorem 1. *The SI-X3DH protocol presented in Section 5 is secure (correct and sound) in the Signal-adapted-CK model of Definition 2, in the random oracle model (where H_1, H_2 and* KDF *are modelled as random oracles), assuming the Honest CDH and Verifiable CDH problems are hard.*

Proof sketch We briefly outline the proof methodology. The proof is similar to the one given by Cohn-Gordon et al. [CGCD$^+$20], adapted to our Signal-adapted-CK model and using the Verifiable and Honest CDH assumptions 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 guessed in advance which session would be tested, and the peer id of that session. It will then simulate the game and insert a VCDH or HCDH challenge into that 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, i.e. there are no collisions. If a key is generated which 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-isogeny) keys, and at most $n + nS$ sending (3-isogeny) keys. A collision among these keys is an instance of the generalized birthday problem:

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

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

So,

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

Where, as before, $|\mathcal{K}_2| = 2^{e_1}$ and $|\mathcal{K}_3| = 3^{e_2}$

Game 2. We guess in advance which session π_u^i the adversary will call the Test() query against, and abort if 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 there exists a session π_v^j which matches π_u^i but $v^* \neq v$. If no such v exists we do not abort. Note too that v is unique even if j is not, so the probability of guessing correctly is $1/n$ and thus:

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

We now take each case in Table 2 separately. We shall have 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, these cases all rely on IK_A and IK_B not being revealed—the adversary should thus be unable to compute SIDH(IK_A, 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 $DH_1 = \text{SIDH}(IK_A, IK_B)$ 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 CDH problem (VCDH) defined above. 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 the correct way 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 both needed to compute the session key but also is the answer to the VCDH challenge. In the second case, the simulator does not know $\mathrm{SIDH}(EK_E, E_B)$ for unknown, potentially malicious ephemeral key EK_E. 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.

In the first case above, if a session key or ephemeral key reveal query is made on such a session, $\mathcal{B}$ returns a random key and maintains a list of such random keys and correspondingly the keys which should have been used to compute it. Then, to ensure that other KDF() queries made are consistent with these replaced keys, we do the following on receipt of a query $\mathrm{KDF}(DH_1 \parallel DH_2 \parallel DH_3)$: $\mathcal{B}$ will query $\mathcal{O}(DH_1)$ and if true, this is exactly the case where $abort_4$ occurs, and $\mathcal{B}$ 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.

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). For each pair of entries (h, h'), we check whether $H_1(\pi_E \oplus h \oplus h')$ is the secret key of EK_E. If such a pair is found, we can use this secret key to compute $\mathrm{SIDH}(EK_E, E_B)$. $\mathcal{B}$ can now use this j-invariant in a query to KDF to compute a consistent session key.

Thus, $\mathsf{Adv}(break_4) = \mathsf{Adv}^{\mathrm{vcdh}}(\mathcal{B})$, 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. Thus, the advantage of winning this game is

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

So 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. The proof is very similar to the first cases above, but now relies on the Honest CDH assumption. 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 $j \in [1, \ldots, m]$, such that signed semi-static key SK_v^j is used in the test session, and aborts if this guess is wrong. Thus,

$$\mathsf{Adv}_3^{1,7} \leq m\mathsf{Adv}_4$$

Game 5 and 6. In game 5, we abort if the adversary queries the KDF oracle with second component DH_2 as the CDH of the test session's EK_u and IK_v. Once again, $\mathcal{B}$ will simulate game 4, but using the received triple (E_A, π, E_B), will replace the ephemeral key of the test session and IK_v with the corresponding E_A and E_B as well as the test session FO-proof with $\pi \oplus \mathrm{SIDH}(E_A, SK_v^j)$.

There are two cases in which $\mathcal{B}$ will not be able to compute valid session keys for non-tested session. The first is for a session where any user initiates with $EK_E \neq EK_u$ to v as the responder. This is because

SIDH(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 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 in the polynomial-length output table of queries $\mathcal{A}$ has made to H_2 all pairs (h, h'). We again check whether $H_1(\pi_E \oplus h \oplus h')$ is the secret key of EK_E. If such a pair is found, we can use the secret key to compute the needed j-invariant SIDH(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 the KDF because $DH_2 = \text{SIDH}(E_A, E_B)$ is unknown. So $\mathcal{B}$ will return a random key and keep a table for consistency as in the previous section. Whenever the adversary makes a query to the KDF oracle, we check if $H_1(\pi \oplus H_2(DH_2))$ corresponds to the secret key of E_A, and if it does, $\mathcal{B}$ has learned DH_2 as the CDH value of E_A and E_B, this is also the case in which the game aborts.

Game 5 is identical to the previous case. So again we have

$$\mathsf{Adv}_3^{1,7} \leq m\mathsf{Adv}^{\text{hcdh}}(\mathcal{B})$$

6.3 Case E_5 (wPFS)

This case relies on EK_A and SK_B not being revealed (assuming 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, and aborts if this guess is wrong. Thus,

$$\mathsf{Adv}_3^5 \leq n_m \mathsf{Adv}_4$$

Game 5 and 6. These proceed exactly as in games 4 and 5 of cases E_1 and E_7 above, but with the challenge key inserted into EK_u and SK_v^j. And 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 $EK_E = EK_u$. This is essentially exactly the same as cases E_1 and E_7. So

$$\mathsf{Adv}_3^5 \leq m\mathsf{Adv}^{\text{hcdh}}(\mathcal{B})$$

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

$$\mathsf{Adv}_{n,m,S}^{\text{exp}} \leq p(n + nm + nS; |\mathcal{K}_2|) + p(n + nS; |\mathcal{K}_3|) + n^2 S \big[\mathsf{Adv}^{\text{vcdh}} + 2m\mathsf{Adv}^{\text{hcdh}}\big]$$

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. □

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 have possibly generated himself—thus it must 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 simulability:

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 IK_B, SK_B, EK_B. The simulator can compute $DH_2 = \mathrm{SIDH}(EK_A, IK_B)$, $DH_3 = \mathrm{SIDH}(EK_A, SK_B)$, and $DH_4 = \mathrm{SIDH}(EK_A, EK_B)$ because x is known, but cannot compute $\mathrm{SIDH}(IK_A, IK_B)$. Under the KDH-type assumptions, there must be an extractor $\hat{\mathcal{B}}$ for Bob's key IK_B—call it $\hat{\mathcal{B}}$. If $\hat{\mathcal{B}}$ outputs $\hat{Z}$ then the shared key is $\mathrm{KDF}(\hat{Z} \parallel DH_2 \parallel DH_3 \parallel DH_4)$—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 value π using the session key it computed. In the second case, no PPT algorithm can compute $\mathrm{SIDH}(IK_A, IK_B)$ without knowing IK_B so the random key is indistinguishable from the real key.

In the case of deniability for the responder, given Bob's public key IK_B, and also a signed semi-static key $SK_B, Sig(SK_B)$. Sim 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 standarization 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 naïve implementation, vulnerable to adaptive attacks), and found it entirely practical.

The SI-X3DH protocol uses three or four SIDH exchanges to form the protocol and derive the shared key—in exactly the same way that Signal X3DH uses three or four DH exchanges. In a single SI-X3DH exchange, the only other information sent (on top of the SIDH keys) is the FO-proof π. This is simply λ 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.

The main drawback 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). These keys also require an SIDH proof of knowledge or honest generation, such as that 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.

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—but this single signature is more efficient than Hashimoto et al. [HKKP21]'s generic scheme which requires two signatures per exchange (one of which must be a more expensive ring signature to achieve deniability).

As mentioned previously, our protocol is more efficient in terms of computation 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 cycles, while Cervantes et al. [COR21] give the SIKEp434 key generation and agreement as 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] can be instantiated using efficient KEMs such as SIKE or other NIST post-quantum KEM candidates, the requirement for a post-quantum secure ring signature is a large drawback to the efficiency of the scheme. Instantiating with the schemes by Beullens et al. [BKP20], choosing the lattice-based instantiation (Falafl) to optimize for speed rather than signature / key size, would be around 78 million clock cycles for signing. Thus the signing time alone is already four times slower than the SI-X3DH key exchange, and such a signature would be around 30KB in size. The smaller isogeny-based instantiation (Calamari) whose signatures are around 3.6KB would take on the order of 10^{11} clock cycles—many orders of magnitude slower.

Thus, concretely, SI-X3DH is the fastest post-quantum alternative to Signal's X3DH 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 use 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 signature on Bob's semi-static keys can use any post-quantum signature scheme, and Bob should additionally register his signature public key so these can be verified.
- When initiating a new key exchange, Alice must also send a small FO-proof (λ-bits) along with her ephemeral public key, and Bob must check this proof on 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 Brendal et al. [BFG+20] were too rushed in dismissing SIDH as a candidate for this reason. While a naïve 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.

[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.

[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.

[GV18] Steven D. Galbraith and Frederik Vercauteren. Computational problems in supersingular elliptic curve isogenies. *Quantum Information Processing*, 17(10):1–22, 2018.

[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.

[MP16] 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.

[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.