

iX3DH: Post-Quantum Subversion-Resilient X3DH Key-Exchange from Isogenies

Tako Boris Fouotsa¹, Trey Li¹, Bernardo Magri¹, and Shancheng Zhang²

¹ The University of Manchester, Manchester, UK
{takoboris.fouotsa,trey.li,bernardo.magri}@manchester.ac.uk

² Beihang University, Beijing, China zscbuaa@buaa.edu.cn

Abstract. Secure messaging runs on commodity devices that can be subverted by malware or supply-chain attacks. *Reverse firewalls* address this threat by interposing a user-side device that mediates the communication of a potentially compromised end-point. Recently, Dodis *et al.* (CRYPTO 2025) proposed a reverse firewall design for protecting Signal, including its Diffie-Hellman based X3DH handshake, against subversion. With Signal and other platforms now migrating to post-quantum cryptography, however, extending this protection to post-quantum, Signal-conforming authenticated key exchanges (AKE) remains a challenge. We present iX3DH, a post-quantum X3DH-style protocol from commutative group-action Diffie-Hellman. Its public group-action values can be re-randomized by a reverse firewall, removing subliminal information from these values. We prove AKE security of iX3DH in the QROM against quantum polynomial-time adversaries, relying on the quantum-accessible group-action gap-CDH assumption, PRF security, and EU-CMA security of a PQ unique signature scheme. We then prove that the reverse firewalls are exfiltration-resistant and preserve security against subversions that maintain functionality and are explainable. To prevent subliminal channels within the signature transcript, we construct IBIS, the first candidate post-quantum unique signature scheme. By integrating IBIS with our firewall-friendly handshake, iX3DH achieves post-quantum AKE security under subversion attacks.

Keywords: Post-quantum cryptography · Isogeny-based cryptography · Key-exchange · Reverse firewalls · Subversion resilience · Unique signatures

1 Introduction

Secure messaging protocols such as Signal protect the confidentiality and authenticity of messages sent over untrusted networks. These guarantees, however, depend on the software at each endpoint implementing the protocol correctly. An attacker who replaces that software through malware or a malicious update can undermine security without breaking the underlying cryptography. For example, the replacement may choose the randomness used to generate an ephemeral key so that its public value leaks secret information, while the key exchange still completes correctly. Such an attack is called a *subversion*.

Cryptographic *reverse firewalls* [35] address this threat by placing a local filter between a potentially subverted client and the network. The filter modifies protocol messages using its own randomness, with the goal of restoring security while preserving functionality. Dodis, Magri, Stephens-Davidowitz, and Tselekounis [22] applied this approach to Signal, protecting both its initial X3DH key exchange and the subsequent messaging protocol. They consider subverted implementations that maintain functionality and are explainable: their behavior must be consistent with an honest execution using some choice of randomness.

That construction relies on classical Diffie-Hellman and therefore does not provide post-quantum security. As secure messaging moves toward post-quantum primitives, its protection against subversion must also be reconsidered. The handshake needs public values that a firewall can rerandomize and signatures that cannot carry subliminal information. We study this problem for the initial key exchange. Our protocol, iX3DH, uses commutative group actions and unique signatures to retain the asynchronous structure of X3DH, with a QROM analysis of the key exchange and a separate security proof for its reverse firewalls.

All authors contributed equally to this work.

1.1 Our Contributions

This work takes a first step toward post-quantum, subversion-resilient secure messaging. Concretely, we make the following contributions.

- *Design a firewall-friendly, isogeny-based variant of X3DH.* Following the blueprint of [22], we construct iX3DH with the same asynchronous publication and delivery flow. Its firewall randomizes ephemeral public values and checks their signatures. We prove exfiltration resistance and AKE security preservation against PPT adversaries whose static stateful subversions maintain functionality and are explainable. The AKE proof simulates the undisclosed subverted participants, then reuses the base reductions (corollary 1).
- *Construct the first candidate post-quantum unique signature scheme.* A challenge in achieving subversion resilience for post-quantum X3DH is to prevent the signature transcript from becoming a subliminal channel, while no post-quantum unique signature scheme was previously known. We address this challenge by constructing IBIS (Isogeny-Based unIque Signature), which is, to the best of our knowledge, the first candidate post-quantum unique signature scheme. In IBIS, uniqueness ensures that, for each verification key and message, there is at most one accepting signature, removing the possibility of leaking information through the choice of signature. Beyond its role in iX3DH, IBIS may be of independent interest.
- *Prove AKE security in the QROM.* We establish the security of iX3DH against quantum adversaries with quantum access to the KDF, under quantum gap-CDH and the stated PRF and signature assumptions (Theorem 2). The result protects session keys against an adversary controlling the server and message delivery, and provides forward secrecy under our exposure model. It preserves the asynchronous setting: Bob can establish a key while Alice is offline.

The base AKE theorem allows a quantum adversary; the firewall result treats classical subversions. The firewalled AKE model permits long-term-key disclosure after acceptance, subject to freshness, while keeping the two subverted participants’ private memories undisclosed. Honest clients retain the base exposure rules. Section 3.3 explains the proof.

Remark 1. Although our construction in Section 3 uses an isogeny-based group action, the AKE reduction applies to any efficiently computable, regular commutative group action on a validated domain for which Q-GA-GDH is hard, together with a secure PRF and a post-quantum signature scheme. Signature uniqueness is additionally needed by the firewall. The source-based freshness and distinct-party Test restrictions apply to each such instantiation.

1.2 Related Work

Our construction brings together reverse firewalls for key exchange and isogeny-based group actions. We discuss these two lines of work and then compare iX3DH with the classical construction of [22].

Subversion-resilient AKE and the post-quantum gap. Cryptographic reverse firewalls (RFs) were introduced by Mironov and Stephens-Davidowitz [35] to restore security against subverted implementations by interposing a user-side sanitizer. Subsequent work designed RFs for various primitives [5,28,16,17,23]. The Signal construction of Dodis *et al.* [22] extends this approach to asynchronous secure messaging, including its initial key exchange. Its use of classical Diffie-Hellman motivates the search for a handshake that also admits post-quantum security.

In the honest-implementation setting, post-quantum adaptations of X3DH are well underway. In practice, Signal has deployed PQXDH [32], a KEM-based upgrade that has undergone extensive formal verification [11]. Theoretically, Hashimoto *et al.* [29] provided a generic post-quantum AKE model, and Brendel *et al.* [13] introduced split KEMs to enable DH-style key reuse without extra rounds. Brendel *et al.* pointed to commutative group actions (GADH) as a natural candidate for this architecture. An explicit isogeny-based adaptation, SI-X3DH [21], was previously proposed but is no longer viable due to key-recovery attacks on its underlying SIDH assumption [14]. These works analyze key exchange with honest implementations; protecting the handshake when client code is replaced requires a subversion model as well.

Duverger *et al.* [26] address subversion-resilient AKE through authenticating and key-securing reverse firewalls. Their model allows different implementation components to be subverted and targets restoration of particular security properties. They introduce rerandomizable KEMs, give Diffie-Hellman and Kyber-based constructions, and combine computational proofs with CryptoVerif verification. Rerandomizable KEMs and commutative group actions thus provide two approaches to handshake messages that a firewall can modify. We use group actions and protect a client whose code may be replaced by a functionality-maintaining, explainable implementation. Our proof treats its local outputs and disclosure queries as well as its messages.

Group action Diffie-Hellman. Following the introduction of CSIDH as a commutative group action analogue of Diffie-Hellman [15], the underlying algebraic structures have seen rapid development. Quantum security margins have been rigorously refined [12,38], and the pursuit of strict constant-time evaluations has led to efficient CSIDH-style algorithms [6,30]. In parallel, a prominent line of work leverages higher-dimensional isogenies to evaluate the class group action for ideals of arbitrary norm in polynomial time, spanning from Clapote(s) [36] to increasingly optimized refinements such as PEGASIS, KLaPoTi, and qt-Pegasis [18,37,19]. While evaluating a single unrestricted group action still takes on the order of a second at cryptographically relevant security levels, even for the fastest of these, these advancements establish GADH as an actively researched and rapidly improving foundation for post-quantum key exchange, motivating our choice to build iX3DH upon it.

Comparison with classical subversion-resilient X3DH. The closest construction to iX3DH is the subversion-resilient X3DH of Dodis *et al.* [22]. We retain its asynchronous publication and delivery flow, using a commutative group action for Diffie-Hellman and IBIS for unique signatures. The main proof difference is the treatment of quantum random-oracle queries: the classical reduction can inspect queries, whereas our reduction must simulate revealed keys and extract DH values without that access. Section 1.3 explains the resulting QROM argument under quantum gap-CDH, including its square-root loss.

Our AKE freshness conditions track every honest source of an accepted ephemeral share, including pending sessions and replays into a different sid. Test concerns distinct parties, and a key reveal from any session with the tested sid invalidates freshness. These conditions restrict the permitted exposures while preserving the asynchronous message flow.

For the firewall, we follow their requirement that subversions maintain functionality and be explainable. We include local AKE results in the comparison: one honest tape must explain both the messages and the session key (Definition 1). We prove exfiltration resistance and, separately, AKE security preservation with the disclosure interfaces of Definition 19. These results concern the key-exchange phase under classical subversion; their construction also treats the subsequent secure-messaging protocol.

1.3 Technical Overview

We next explain how group-action rerandomization and unique signatures support the firewall, and how the AKE reduction handles quantum queries.

Group-action key exchange. We first recall the group-action analogue of Diffie-Hellman. Let cl be a finite abelian group acting freely and transitively on a finite set $\mathcal{E}$ by an efficiently computable action $\star$, and let $E \in \mathcal{E}$ be a public base object. The group action Diffie-Hellman protocol proceeds as follows. Alice samples $[a] \leftarrow \text{cl}$ and sends $A = [a] \star E$. Bob samples $[b] \leftarrow \text{cl}$ and sends $B = [b] \star E$. Alice computes $[a] \star B = [a] \star ([b] \star E) = ([a][b]) \star E$, while Bob computes $[b] \star A = [b] \star ([a] \star E) = ([b][a]) \star E$. Since cl is abelian, both parties obtain the same shared set element $([a][b]) \star E \in \mathcal{E}$. This is the direct analogue of the classical Diffie-Hellman computation in which Alice and Bob exchange g^a and g^b and derive g^{ab} .

Re-randomization of group-action public keys. The key property that makes group actions useful for reverse firewalls is that public group-action elements can be re-randomized. Given a public element $A = [a] \star E$, a firewall can sample $[r] \leftarrow \text{cl}$ and compute $[r] \star A = [r] \star ([a] \star E) = ([r][a]) \star E$. For every fixed $[a]$, if $[r]$ is uniform over cl , then $[r][a]$ is also uniform over cl . Thus the re-randomized value $([r][a]) \star E$ has the same distribution as a freshly generated honest group-action public key. This is the group-action counterpart of re-randomizing a classical Diffie-Hellman value g^a into g^{ar} .

This re-randomization is precisely what a reverse firewall needs. If a subverted implementation tries to encode subliminal information in the outgoing public value $[a] \star E$, the firewall replaces it by a re-randomized value $[r] \star ([a] \star E)$ before it reaches the peer. At the same time, we require that the firewall preserves correctness: the peer should still derive the same session key.

Isogeny-based unique signature. Re-randomizing the group action transcript is not sufficient by itself. The protocol also signs ephemeral public keys, and signatures may provide a subliminal channel if a subverted implementation can choose among several valid signatures on the same message. For this reason, iX3DH uses a post-quantum unique signature scheme. Uniqueness means that, for every verification key and message, there is at most one accepting signature. This ensures that the protocol has no remaining freedom to encode information in the signature.

Our Isogeny-Based unIQue Signature scheme IBIS is built upon the DeuringVUF based hash-and-sign signature [34]. In this signature, one hashes the message to a kernel $\langle K \rangle$ of large prime order q , and the signature is a representation of the isogeny $\phi : E \rightarrow E'$ of degree q and of kernel $\langle K \rangle$. The signer relies on the fact that they know the endomorphism ring of E when computing ϕ , as computing isogenies of large prime degree is hard otherwise. Moreover, the efficient representation [40] of isogenies of large prime degree, namely the dimension-two representation, makes use of an auxiliary isogeny $\psi : E \rightarrow E''$. This auxiliary isogeny is random and is part of the signature, preventing it from being unique. In the design of IBIS, we detach this auxiliary isogeny from the signature. Since it is independent of the isogeny ϕ and can be generated from $(E, \text{End}(E))$ and q which the signer generates during key generation, ψ can also be generated at the same time and appended to the public key. Once this is done, we are left to determining a way to verifiably uniquely describe the actual signature ϕ . For this, we notice that after canonicalising the end curve E' , the set of all possible valid signatures ϕ is finite and has cardinality 2 most of the time, and exceptionally 4 or 6. We equip this set with a strict total order, and ask for the signature to be the minimum of the set. The verifier after verifying the signature as they would, they also check that the curve E' is the canonical one and that ϕ is the minimum element of the set of all possible valid signatures.

The DeuringVUF of [34] gives a unique output for each public key and input. Output uniqueness alone does not make its proof unique: IBIS also fixes the auxiliary data and the representation of the response isogeny, as described above. Applying this approach to CSI-FiSh [10], SQIsign [1] or WaterSQI [27] would require a way to select and verify a canonical signing transcript. For hash-and-sign schemes such as PRISM [7] and PRISMO [27], selecting a canonical response among exponentially many valid isogenies also appears challenging.

Quantum random oracle reduction. In a classical random-oracle proof, the reduction can record a hash query and later recognize the DH values used by a session. A quantum query may be a superposition of many inputs. Reading it changes the adversary's state and can change which session it eventually tests. There is also a simulation problem before extraction: iX3DH has no explicit key confirmation, and the adversary can reveal another session's key without first providing its DH values in a classical query. The reduction must answer that reveal consistently even when it does not know the session's DH secrets. Asynchronous publication and replays mean that a session carrying a challenge share can participate in several such exchanges.

The KDF takes sid and the three DH values as input. The simulator can therefore associate a random seed with each sid before any session completes. An unrestricted quantum DDH oracle recognizes the unique valid DH triple for that sid , so revealed keys agree with all earlier quantum queries. Sources may be pending, replayed, or involved in other exchanges.

The test transcript can depend on earlier hash queries. We therefore prove a query-extraction bound that obtains an earlier DH triple together with a compatible later transcript and its randomizers. Ordinary O2H handles queries after Test. The resulting bound has a square-root loss in the gap-CDH advantage. The theorem covers exchanges between distinct parties under the source-based exposure rules. Its proof sketch follows Theorem 2.

Computational assumptions. The AKE reduction uses Q-GA-GDH, PRF security against quantum distinguishers with classical queries, and post-quantum existential unforgeability with classical signing queries. The gap assumption explicitly permits coherent DDH tests on unrestricted triples, as specified in Definition 13. The reverse firewall uses rerandomization of the group action and signature uniqueness. Our isogeny-based signature is based on the One-More Isogeny Problem (OMIP), stated as Problem 1.

1.4 Organization

Section 2 gives the cryptographic building blocks, reverse-firewall definitions and group-action assumptions. Section 3 presents iX3DH, its firewalls, and the statements and proof sketches of both security results. Section 4 constructs IBIS and proves uniqueness. Appendix A contains the full correctness and AKE proofs, including the simulation for subverted clients. Appendix B specifies the base and firewalled AKE experiments.

2 Preliminaries

2.1 Notations

We denote ideals in Fraktur, such as $\mathfrak{a}, \mathfrak{b}, \mathfrak{x}, \mathfrak{y}, \alpha, \beta$, with fraktur Greek letters used analogously to the Roman ones. Ideal classes are written in square brackets, e.g., $[\mathfrak{a}], [\mathfrak{b}], [\mathfrak{x}], [\mathfrak{y}], [\alpha], [\beta]$. We also use the standard elliptic-curve notation $[\alpha]P$ for scalar multiplication of a point P by an integer or residue class $\alpha \in \mathbb{Z}/q\mathbb{Z}$. The action of an ideal class on an elliptic curve is denoted by $\star$, for example $[\mathfrak{a}] \star E$. We use letters from the beginning of the alphabet (e.g., $[\mathfrak{a}], [\mathfrak{b}]$) for long-term keys, and letters from the end of the alphabet (e.g., $[\mathfrak{x}], [\mathfrak{y}]$) for ephemeral keys.

We write $a \leftarrow \mathcal{A}$ to denote the output of a (possibly randomized) algorithm $\mathcal{A}$, and $s \leftarrow S$ to denote uniform sampling of s from a set S . We write $A \times B$ to denote the Cartesian product of sets A and B . We sometimes use $[a]$ to denote the integer set $\{1, \dots, a\}$ for compactness, where a is a positive integer. Furthermore, as our protocol is game-based, we use several related notations. We denote by $\text{Adv}_F^C(\mathcal{A})$ the *advantage* of an adversary $\mathcal{A}$ against a construction C in an unpredictability game F . The notation **req** is a requiring oracle to judge a special condition.

We denote tensor product by $\otimes$ and bit-wise XOR by $\oplus$. For quantum random oracle notations, we use Dirac notation for computational basis states. In particular, $|x, z\rangle$ denotes $|x\rangle \otimes |z\rangle$, where $x \in \{0, 1\}^n$ is a bit string stored in the first (query) register, and $z \in \{0, 1\}^m$ is a bit string stored in the second (auxiliary) register.

2.2 Reverse Firewall Model

We use the framework of Mironov and Stephens-Davidowitz [35]. A reverse firewall W sits between a party P_1 and the network and modifies its incoming and outgoing messages. We write $W \circ P_1$ for their composition and $\Pi_{P_1 \rightarrow \tilde{P}_1}$ for a protocol in which P_1 is replaced by code $\tilde{P}_1$.

Following [35, Definition 2(5)], $\tilde{P}_1$ is functionality maintaining if, for every fixed valid input, $\Pi_{P_1 \rightarrow \tilde{P}_1}$ satisfies the output requirements $\mathcal{F}$ except with negligible probability over setup and execution randomness. Following [22], we additionally require explainability: the replacement's behavior must be consistent with some choice of honest randomness. For AKE, this comparison includes the local session results as well as the messages.

Definition 1 (Explainable implementations). Let $P(I, m_1, \dots, m_t; r)$ denote the messages and local results produced by party P with input I and random tape r on successive invocations $m_1, \dots, m_t$. The input fixes the registered keys and public parameters. Invocations include session starts and received messages, together with their session indices; they may interleave sessions and contain invalid messages. The record includes each invocation's outgoing messages, empty response or abort, and the fields of every session: role role , peer Pid , identifier sid , acceptance status Ψ and key $\mathbf{k}$, with $\mathbf{k} = \perp$ before acceptance.

A possibly stateful PPT implementation $\tilde{P}$ is explainable if, for every valid input I , every polynomially bounded invocation sequence $m_1, \dots, m_t$, and every random tape r' , there exists a random tape r such that

$$P(I, m_1, \dots, m_t; r) = \tilde{P}(I, m_1, \dots, m_t; r').$$

The equality holds for the entire sequence under one tape r . It is required for every fixed choice of the random-oracle functions, shared by both executions. Any auxiliary information of the replacement is fixed in its initialization. Private memory and private oracle-query transcripts are not part of the compared record, and the record does not give the adversary additional access to session keys.

This is the formulation of [22, Definition 18], extended to include local AKE results. It applies to the client on the inputs actually delivered by its firewall. The explaining tape need not be sampled from the honest distribution or efficiently recovered. Random choices may therefore be biased, depend on private keys, or be correlated across sessions. Messages and session keys must nevertheless have the same honest explanation.

Definition 2 (Reverse Firewall [23]). *A reverse firewall W maintains functionality $\mathcal{F}$ for party P_1 in protocol Π if Π satisfies $\mathcal{F}$ and, for every polynomially bounded $k \geq 1$, the protocol $\Pi_{P_1 \rightarrow W^k \circ P_1}$ also satisfies $\mathcal{F}$, where W^k denotes k successive firewall instances.*

Relation to the model of [22]. We follow [22, Definitions 18–19] in requiring subverted implementations to be functionality maintaining and explainable. Explainability requires their behavior on every incoming sequence to have an honest explanation, including when the peer sends invalid messages. In Definition 1, we extend their outgoing-message comparison to the local AKE results, using one random tape for both. The explaining randomness may be biased and correlated across sessions.

The exfiltration game compares honest and subverted implementations behind the same firewall, following [35]. It does not provide secret-state disclosures or a Test interface. Corollary 1 supplies a separate AKE security-preservation proof for these implementations under the disclosure rules of Definition 19. We study the asynchronous key-exchange phase rather than the full secure-messaging experiment of [22].

2.3 Exfiltration Resistance and Security Preservation

We use the notions of Mironov and Stephens-Davidowitz [35]. Adversaries and replacement implementations in this subsection are PPT. Exfiltration resistance compares a subverted implementation with an honest implementation, both protected by the same firewall.

Definition 3 (Exfiltration Resistance [35, Definition 6]). *A reverse firewall W is strongly exfiltration-resistant for P_1 against P_2 in protocol Π if every PPT adversary $\mathcal{A}$ has negligible advantage $|\Pr[\text{LEAK} = 1] - 1/2|$ in the following game. It is weakly exfiltration-resistant if this holds for every such adversary whose replacement $\tilde{P}_1$ maintains functionality $\mathcal{F}$.*

Exfiltration Resistance Game

```

LEAK( $\lambda, \Pi, P_1, P_2, W$ ):
( $\sigma_{\mathcal{A}}, \tilde{P}_1, P_2^*, I$ )  $\leftarrow \mathcal{A}(1^\lambda)$ 
 $b \leftarrow \{0, 1\}$ 
if  $b = 1$  then
   $P^* \leftarrow W \circ \tilde{P}_1$ 
else
   $P^* \leftarrow W \circ P_1$ 
end if
 $T \leftarrow \Pi_{P_1 \rightarrow P^*, P_2 \rightarrow P_2^*}(I)$ 
 $b' \leftarrow \mathcal{A}(\sigma_{\mathcal{A}}, T, \sigma_{P_2^*})$ 
return ( $b = b'$ )

```

Here I is a valid protocol input, T is the protocol transcript, $\sigma_{\mathcal{A}}$ is the adversary's retained state, and $\sigma_{P_2^*}$ is the adversarial peer's final state. Several adversarial peers can be represented by the single party P_2^* .

Definition 4 (Security Preservation [35, Definition 5]). *Let Π satisfy security requirements $\mathcal{S}$ and functionality requirements $\mathcal{F}$. A reverse firewall W strongly preserves $\mathcal{S}$ for P_1 if $\Pi_{P_1 \rightarrow W \circ \tilde{P}_1}$ satisfies $\mathcal{S}$ for every PPT replacement $\tilde{P}_1$. It weakly preserves $\mathcal{S}$ if this holds for every functionality-maintaining replacement.*

For AKE, $\mathcal{S}$ specifies the security game, including its queries and freshness conditions. Exfiltration resistance alone does not account for state disclosures and Test replies, so we prove AKE security preservation separately. Our results concern functionality-maintaining, explainable implementations in the sense of Definition 1, with the AKE disclosure rules stated in Definition 19.

2.4 Cryptographic Building Blocks

In this section, we recall the basic building blocks used in this work.

Signature schemes. A signature scheme $\Sigma = (\text{KeyGen}, \text{Sign}, \text{Ver})$ consists of the following polynomial-time algorithms:

- $\text{KeyGen}(1^\lambda) \rightarrow (\text{ssk}, \text{spk})$ outputs a signing key ssk and a verification key spk .
- $\text{Sign}(\text{ssk}, m) \rightarrow \sigma$ outputs a signature σ on message m .
- $\text{Ver}(\text{spk}, m, \sigma) \rightarrow \{0, 1\}$ is deterministic and outputs 1 iff σ verifies for m under spk .

The standard security requirement for such a scheme is existential unforgeability against chosen-message attacks.

Definition 5 (EU-CMA Security). A signature scheme Σ is post-quantum EU-CMA secure if for all QPT adversaries $\mathcal{A}$:

$$\Pr \left[\begin{array}{c} \text{Ver}(\text{spk}, m, \sigma) = 1 \\ \wedge m \notin Q \end{array} \middle| \begin{array}{c} (\text{ssk}, \text{spk}) \leftarrow \text{KeyGen}(1^\lambda); \\ (m, \sigma) \leftarrow \mathcal{A}^{\text{Sign}(\text{ssk}, \cdot)}(\text{spk}) \end{array} \right] \leq \text{negl}(\lambda),$$

where Q is the set of messages queried by $\mathcal{A}$ to the signing oracle $\text{Sign}(\text{ssk}, \cdot)$. Signing queries and their answers are classical; $\mathcal{A}$ may otherwise perform quantum computation.

While EU-CMA security protects against external forgeries, a subverted signer already holds the signing key. If it can efficiently compute several accepting signatures on one message, their choice can carry information across the firewall. Making the honest signing algorithm deterministic does not by itself remove this choice: replacement code can use another algorithm that produces signatures accepted by the same verifier. Conversely, the mere existence of several valid signatures does not imply that the signer can efficiently find them. The property our proof uses is uniqueness of the accepting signature for each fixed verification key and message.

Signature *uniqueness* removes the signature-choice channel studied in [5]: once the firewall fixes the message, any different signature fails verification. The signer can still withhold its response or alter other protocol behavior. Explainability supplies the required message and abort behavior, while uniqueness fixes the valid signature on each prescribed message.

Definition 6 (Uniqueness). Let Σ be a signature scheme. We say that Σ satisfies uniqueness if for all $\lambda \in \mathbb{N}$, for all spk output by $\text{KeyGen}(1^\lambda)$, and all messages m , there exists a single value σ such that $\text{Ver}(\text{spk}, m, \sigma) = 1$.

Pseudorandom functions. Next we recall the standard definition of pseudorandom functions (PRF).

Definition 7 (Pseudorandom Function). Let $\text{PRF} : \{0, 1\}^\lambda \times \mathcal{X} \rightarrow \mathcal{Y}$ be an efficient keyed function, where $\mathcal{X}$ and $\mathcal{Y}$ denote the domain and the range of PRF. Denote by $\mathcal{F}$ the set of all functions mapping $\mathcal{X}$ into $\mathcal{Y}$. Intuitively, a pseudorandom function (PRF) is a function that is computationally indistinguishable from a truly random function, for all efficient distinguishers. More formally, a function $\text{PRF} : \{0, 1\}^\lambda \times \mathcal{X} \rightarrow \mathcal{Y}$ is a (t, q, ε) -secure pseudorandom function, if for all adversaries $\mathcal{A}$ running in time at most t we have

$$\left| \Pr[\mathcal{A}^{\text{PRF}_k(\cdot)}(1^\lambda) = 1] - \Pr[\mathcal{A}^{f(\cdot)}(1^\lambda) = 1] \right| \leq \varepsilon,$$

where the randomness is taken over the choice of k and f , and $\mathcal{A}$ asks at most q queries to its oracle.

We require security against QPT distinguishers with classical queries to the challenge oracle. This is the convention used by the PRF reductions in Theorem 2 and Corollary 1.

2.5 Quantum Random Oracle

We model the KDF as a function $H : \{0, 1\}^* \rightarrow \{0, 1\}^\lambda$ sampled uniformly once. Honest inputs canonically encode a session identifier and a triple from $\mathcal{E}^3$. Quantum queries apply

$$U_H : |x, z\rangle \mapsto |x, z \oplus H(x)\rangle.$$

All honest and adversarial queries use this same function. A classical table that samples an undefined entry during a superposition query does not specify this unitary.

For a polynomial bound Q on the total number of queries, a fixed $2Q$ -wise independent function has exactly the same observable behavior [44]. This bound counts queries by the adversary and by the simulation. If an extractor invokes or reverses the simulation, its additional queries must also be counted when choosing the independence parameter. Input lengths are bounded by the running time and can be injectively encoded into a finite domain.

One-way to hiding. Let (G, H, S, ρ) have a joint distribution, where G, H agree outside S and ρ is the algorithm's initial state. For an algorithm making at most q oracle calls, the fixed-function hybrid argument bounds its distinguishing advantage by $2q\sqrt{p_{\text{guess}}}$, where p_{guess} is the probability that measuring a uniformly chosen query input in the prescribed execution yields an element of S . This is the argument underlying ordinary O2H [4, Theorem 3]. All calls made by the adversary and the surrounding simulation count. We use its common-initial-state form and give the hybrid calculation in Appendix A.2; the state may be correlated with the two functions.

The functions must be fixed for this application. In our proof they are fixed at Test, and O2H handles the subsequent queries. Queries before Test require a separate argument: measuring one of them does not necessarily preserve the later transcript that determines the target input. We give that argument explicitly in Appendix A.2.

2.6 Group Action Problems

Let λ be the security parameter, and recall from Section 2.1 that cl denotes an ideal class group acting via $\star$ on a finite set $\mathcal{E}$ of elliptic curves, possibly equipped with auxiliary orientation data. For every $[a], [b] \in \text{cl}$ and every $E \in \mathcal{E}$, we have $1_{\text{cl}} \star E = E$ and $[a] \star ([b] \star E) = ([a][b]) \star E$. Since cl is abelian, we have $[a] \star ([b] \star E) = [b] \star ([a] \star E)$. As cl is an ideal class group, the product in cl may equivalently be written as $[a][b] = [ab]$. Next we define the counterparts of the classical problems.

Analogous to the classical problems of discrete logarithm, where given (g, g^x) one is asked to find x , the group action inverse problem (GAIP) is defined as follows.

Definition 8 (GAIP). *Given $(E, [x] \star E)$ for $[x] \leftarrow \text{cl}$, find a representation of $[x]$.*

Analogous to the classical computational Diffie-Hellman problem, where, given (g, g^x, g^y) for random x, y , one is asked to compute g^{xy} , the group-action computational Diffie-Hellman (GA-CDH) problem is defined as follows.

Definition 9 (GA-CDH). *Given $(E, [x] \star E, [y] \star E)$ for independent $[x], [y] \leftarrow \text{cl}$, compute $([x][y]) \star E$.*

Analogous to the classical decisional Diffie-Hellman problem, where one is asked to distinguish $D_0 = (g, g^x, g^y, g^{xy})$ from $D_1 = (g, g^x, g^y, g^z)$ for random x, y, z , the group action decisional Diffie-Hellman (GA-DDH) problem is defined as follows.

Definition 10 (GA-DDH). *Distinguish the distribution $D_0 = (E, [x] \star E, [y] \star E, ([x][y]) \star E)$ from $D_1 = (E, [x] \star E, [y] \star E, [z] \star E)$, where $[x], [y], [z] \leftarrow \text{cl}$ are sampled independently.*

Analogous to the classical division-computational Diffie-Hellman problem, where given (g, g^x, g^y) one is asked to compute $g^{x/y}$, the class group division-computational Diffie-Hellman (GA-DCDH) problem is defined as follows.

Definition 11 (GA-DCDH). *Given $(E, [x] \star E, [y] \star E)$ for independent $[x], [y] \leftarrow \text{cl}$, compute $([x][y]^{-1}) \star E$.*

Analogous to the classical gap-Diffie-Hellman, where given a DDH oracle one is asked to solve CDH, the group action gap-Diffie-Hellman problem is GA-CDH with the adversary additionally given access to a GA-DDH oracle.

For the fixed public base E , let $D_E(U, V, W) = 1$ exactly when $U = [u] \star E$, $V = [v] \star E$, and $W = [uv] \star E$ for classes $[u], [v] \in \text{cl}$. The oracle evaluates this predicate exactly on every valid triple and returns zero on invalid encodings.

Definition 12 (GA-GDH). *Given $(E, [x] \star E, [y] \star E)$ for independent uniform $[x], [y] \in \text{cl}$, compute $[xy] \star E$, with classical oracle access to D_E on arbitrary triples.*

Definition 13 (Q-GA-GDH). *The challenge and target are those of GA-GDH, but the algorithm is QPT and has coherent access to*

$$|U, V, W, z\rangle \mapsto |U, V, W, z \oplus D_E(U, V, W)\rangle.$$

All three decision arguments may vary in superposition. We denote the probability that $\mathcal{B}$ outputs the correct computational target by

$$\text{Adv}_{\text{Q-GA-GDH}}(\mathcal{B}).$$

We assume hardness of Q-GA-GDH. This implies hardness of GA-CDH, since a computational solver can ignore the decision oracle. The classical group-action gap assumption is used, for example, in [2]. Duman *et al.* [25] study quantum-accessible variants with a fixed decision argument. Our assumption permits all three arguments to vary coherently; its hardness is an explicit premise beyond classical or fixed-argument decision access.

3 iX3DH: Isogeny-Based Subversion-Resilient X3DH

We describe iX3DH, our isogeny-based firewall-friendly protocol (Fig. 1), and prove its AKE security in the model of Appendix B. Section 3.3 gives the reverse firewalls and proves exfiltration resistance and AKE security preservation under explicit restrictions on subverted clients and their memory disclosures.

3.1 Protocol Description

We now describe the construction of our protocol, as detailed in Fig. 1. The protocol follows the firewall-friendly X3DH framework of [22], but replaces classical Diffie-Hellman operations with a commutative group action foundation. We first specify the cryptographic primitives and public parameters used throughout the protocol, and then describe the long-term key generation procedure and the session key establishment phase.

Cryptographic building blocks. Our construction relies on three primary building blocks. First, a post-quantum unique signature scheme $\Sigma = (\text{KeyGen}, \text{Sign}, \text{Ver})$. Second, a pseudorandom function family $\text{PRF} : \{0, 1\}^\lambda \times \{0, 1\}^* \rightarrow \{0, 1\}^\lambda$, where $\{0, 1\}^\lambda$ denotes the key space. We require security of PRF against QPT adversaries making classical oracle queries. Finally, a key derivation function $H : \{0, 1\}^* \rightarrow \{0, 1\}^\lambda$. We model H as a quantum-accessible random oracle. Its inputs are the session identifier sid and the three GADH shared secrets. Quantum access applies the unitary $|x, z\rangle \mapsto |x, z \oplus H(x)\rangle$ for a fixed random function H , independent of any oracle used by the signature scheme.

Public parameters. The public parameters fix the commutative group-action setting used by all parties. They consist of an ideal class group cl , a finite set $\mathcal{E}$ of elliptic curves, a free and transitive, efficiently computable action $\star : \text{cl} \times \mathcal{E} \rightarrow \mathcal{E}$, and a public base object $E \in \mathcal{E}$. They also specify canonical injective encodings for public group-action elements, protocol transcripts and inputs to H . Tuple arguments to functions use these encodings implicitly. Public keys and received action elements are validated in $\mathcal{E}$. Server randomizers are represented classes in cl supporting efficient action, multiplication and inversion; the server may choose any such class, including the identity, adaptively. Malformed inputs are rejected.

iX3DH protocol

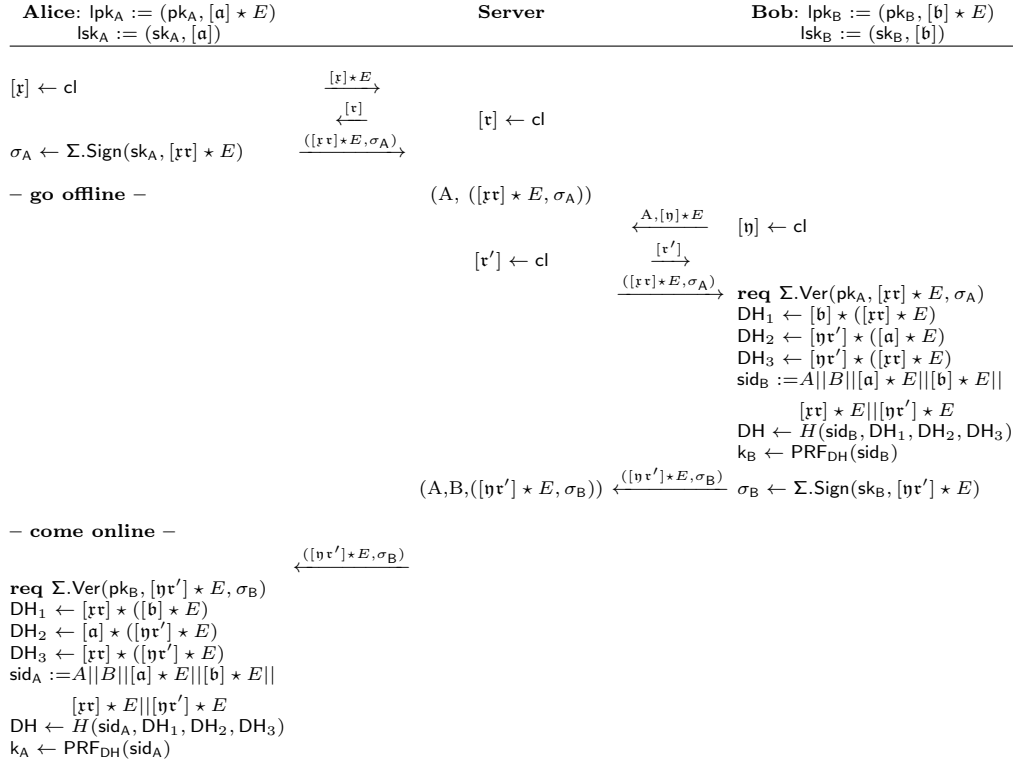


Fig. 1. Isogeny-based Firewall-friendly iX3DH protocol.

Long-term keys. Each party generates a signing key pair together with a long-term group-action secret. Concretely, Alice samples

$$(\text{sk}_A, \text{pk}_A) \leftarrow \Sigma.\text{KeyGen}(1^\lambda) \quad \text{and} \quad [\mathbf{a}] \leftarrow \text{cl},$$

where $[\mathbf{a}]$ is sampled uniformly at random, and sets

$$\text{lsk}_A = (\text{sk}_A, [\mathbf{a}]), \quad \text{lpk}_A = (\text{pk}_A, [\mathbf{a}] \star E).$$

Bob analogously samples

$$(\text{sk}_B, \text{pk}_B) \leftarrow \Sigma.\text{KeyGen}(1^\lambda) \quad \text{and} \quad [\mathbf{b}] \leftarrow \text{cl},$$

where $[\mathbf{b}]$ is sampled uniformly at random, and sets

$$\text{lsk}_B = (\text{sk}_B, [\mathbf{b}]), \quad \text{lpk}_B = (\text{pk}_B, [\mathbf{b}] \star E).$$

The public keys are distributed to the other participants before the key-exchange session. In the figure, A and B denote the initiator's and responder's identities, whose public keys are fixed by the registry.

Protocol execution. We next describe how Alice and Bob establish a session key. The main difference from the Diffie–Hellman-based construction of [22] is that all key-agreement values are computed through the group action of cl on $\mathcal{E}$, as illustrated in Fig. 1. The server is used only to assist the firewall-friendly flow: it stores signed ephemeral material and supplies fresh randomizers for the parties' ephemeral keys. The description below is an honest execution; the security experiment allows the server to choose randomizers adaptively.

Alice begins by sampling an ephemeral secret $[\mathbf{x}] \leftarrow \text{cl}$ and sending the corresponding public value $[\mathbf{x}] \star E$ to the server. The server samples a random ideal class $[\mathbf{t}] \leftarrow \text{cl}$ and returns it to Alice. Alice

then derives the randomized ephemeral value $[\mathbf{r}\mathbf{r}] \star E$, signs it using $\mathbf{sk}_A$, and uploads the resulting signature to the server. This signed randomized key can later be retrieved by Bob even if Alice is offline. Bob performs the same preparation step with his own ephemeral secret $[\mathbf{r}] \leftarrow \text{cl}$ and a server-provided randomizer $[\mathbf{r}'] \leftarrow \text{cl}$. In particular, Bob obtains the randomized value $[\mathbf{r}\mathbf{r}'] \star E$, signs it using $\mathbf{sk}_B$, and stores the signature on the server.

When Alice reconnects, she downloads Bob’s randomized ephemeral key together with its signature. After verifying the signature under Bob’s public verification key, Alice computes the three GADH-derived values

$$[\mathbf{r}\mathbf{r}] \star ([\mathbf{b}] \star E), [\mathbf{a}] \star ([\mathbf{r}\mathbf{r}'] \star E), [\mathbf{r}\mathbf{r}] \star ([\mathbf{r}\mathbf{r}'] \star E).$$

These values and the session identifier sid_A are used to derive the intermediate key material

$$\text{DH} := H(\text{sid}_A, \text{DH}_1, \text{DH}_2, \text{DH}_3).$$

Finally, Alice derives the session key by applying PRF to the session identifier: $k_A := \text{PRF}_{\text{DH}}(\text{sid}_A)$. Bob carries out the analogous verification and group-action computations from his side, obtains the same value DH , and outputs $k_B := \text{PRF}_{\text{DH}}(\text{sid}_B)$. By correctness of the group action, the two parties obtain matching session keys. Bob accepts when he sends his response, and Alice accepts when she processes it. Pending sessions retain their ephemeral secrets; completed sessions erase them as specified in Appendix B.

3.2 Security

We analyze iX3DH in the QROM using the model in Appendix B. Partnering requires agreement on the session identifier. The source relation protects the ephemeral shares used by a session even when their publishing sessions are pending or have different session identifiers. These restrictions weaken the exposure guarantee relative to [29,22], while preserving the asynchronous message flow.

Theorem 1. *If the signature scheme Σ is perfectly correct, then the protocol given by Fig. 1 is 1-correct.*

The proof of Theorem 1 is deferred to Appendix A.1.

The bound below uses the quantum gap-CDH assumption of Definition 13, the construction in Fig. 1, and the source-based freshness rules. Test is restricted to exchanges between distinct parties; the experiment still allows other sessions to run. Here “distinct” refers to party identities. Partners must have the same defined sid , and replays may create several partners or same-owner copies of one key. The model excludes key reveals from all such copies and does not assume partner uniqueness.

Theorem 2. *Let $\mathcal{A}$ be a QPT adversary making at most q_s quantum KDF queries in $G_{\Pi_{\text{AKE}}}^{\text{FS}}(\mu, \ell)$, and put $N = \mu\ell$ and $Q = q_s + N + 1$. There are six QPT Q-GA-GDH reductions, three QPT PRF reductions with classical challenge queries, and one QPT EU-CMA reduction with classical signing queries. Let ϵ_g , ϵ_p , and ϵ_{sig} denote their respective maximum advantages. Then*

$$\text{Adv}_{\Pi_{\text{AKE}}}^{\text{AKE-FS}}(\mathcal{A}) \leq (2N^2 + N\mu)(6Q\sqrt{\epsilon_g} + \epsilon_p) + \frac{1}{2}\mu\epsilon_{\text{sig}}.$$

Consequently, iX3DH is AKE-secure in this model under Q-GA-GDH, PRF security, and signature unforgeability as specified above.

Proof sketch. The proof must identify a hidden DH component and simulate the rest of the execution consistently with every quantum KDF query. The first task depends on which secrets remain hidden; the second must also cover pending sessions, replays, and adversarially chosen randomizers. We explain both below; Appendix A.2 gives the full reductions.

First, an honest source of an accepted share need not be a partner. Bob can accept while Alice is offline, and Alice’s signed share can be replayed into another exchange. We therefore track every honest session of the peer that published the accepted share before acceptance, regardless of its role, sid , or completion status. Freshness protects these sources as well as the target. Test concerns distinct parties, so a source and the target occupy different sessions and their raw ephemeral secrets are sampled independently. It also excludes a key reveal from any session with the target sid : equal session identifiers give equal DH tuples and keys, even for two copies owned by one party. If the accepted share has no

honest source, freshness keeps the peer uncorrupted until acceptance. The accepted signature is then a forgery on a message that the peer never signed.

With an honest source, freshness leaves three cases. If neither the target's state nor any source state was exposed, we embed the two challenge secrets in the target and one source ephemeral key. Their ephemeral-ephemeral DH share solves the challenge after removing the server randomizers. If the target state was exposed, its owner's long-term key and every source state remain hidden; we embed there and use the corresponding mixed DH share. If a source state was exposed, the target state and the peer's long-term key remain hidden, giving the other mixed embedding. Exposing both ephemeral sides is excluded by freshness. The reductions stop before a reveal of an embedded secret; these stops cannot remove a correctly guessed execution in its selected fresh case.

For example, in the first case the hidden raw secrets are $[\alpha], [\beta]$ and the valid third share is $w_3 = [\alpha\beta\mathbf{r}_{\mathfrak{T}}\mathbf{r}_{\mathfrak{S}}] \star E$. Once the reduction knows the delivered randomizers $[\mathbf{r}_{\mathfrak{T}}], [\mathbf{r}_{\mathfrak{S}}]$, it recovers the challenge target by applying $[\mathbf{r}_{\mathfrak{T}}\mathbf{r}_{\mathfrak{S}}]^{-1}$ to w_3 . The mixed cases remove one randomizer in the same way.

To simulate the KDF, write $c = (I, J, A, B, X, Y)$ for the sid and w for the three DH values. Here I, J are the ordered identities, A, B the long-term action public keys, and X, Y the final ephemeral elements. Regularity gives exactly one valid triple $w(c)$ for each c . Coherent DDH queries recognize this triple. Take independent random functions R on session identifiers and $\mathcal{J}$ on full KDF inputs, and answer

$$H(c, w) = \begin{cases} R(c), & w = w(c), \\ \mathcal{J}(c, w), & w \neq w(c). \end{cases}$$

This is exactly a random function: every valid input uses its own independent R entry, and every other input uses its own independent $\mathcal{J}$ entry. An honest session's seed is consequently $R(c)$, which the simulator can evaluate without knowing its DH secrets. This equality holds from the start, including for other exchanges of a chosen source. Since messages do not depend on the session key, its internal evaluation can be deferred until Test or a key reveal. Recording each key reveal before evaluating R rules out an honest internal call at the tested sid in a permitted Test prefix. Different session identifiers use independent seeds even if their DH triples coincide, so no global collision abort is needed.

This oracle is implemented reversibly. On each basis input (c, w) , the simulator computes the three DDH decision bits, uses their conjunction to select R or $\mathcal{J}$, and uncomputes the bits. It therefore acts coherently on a superposition of candidate triples. The check recognizes a valid triple supplied in a query without computing the hidden DH values itself. Computing and uncomputing the bits uses six decision-oracle calls per KDF query.

The remaining difficulty is timing. The adversary can query the valid triple before delivering the randomizers to the selected sessions. Measuring such a query gives the DH values, but the reduction still needs those randomizers to recover the challenge solution. Our extraction argument therefore keeps a continuation. It measures one randomly chosen call to R , failing if its validity control is inactive. It answers that call normally or with zero, chosen by a fair bit, and answers all later queries at the measured sid with zero. It succeeds only if this modified execution reaches a permitted Test with the same sid and supplies the required randomizers. No claim is made that measurement preserves the original continuation.

The quantitative bound uses the Fourier representation of the oracle. Let p be the weight, just before Test, of permitted executions whose tested oracle cell is nonzero; this records dependence on that cell through earlier queries. Writing p_{ext} for the extractor's success probability, a telescoping argument gives $p \leq 4Q^2 p_{\text{ext}} \leq 4Q^2 \epsilon_g$. Swapping that cell with a fresh independent cell changes any output probability by at most $2\sqrt{p}$, hence costs at most $4Q\sqrt{\epsilon_g}$. After Test, all required randomizers are known. Ordinary O2H then bounds the effect of restoring the original oracle while retaining the independent Test seed by $2Q\sqrt{\epsilon_g}$.

Finally, PRF security replaces the Test key with a uniform value. This reduction samples every action secret itself, so it needs no DDH oracle. Each guessed DH case costs at most $6Q\sqrt{\epsilon_g} + \epsilon_p$. Guessing the target and its earliest source costs N^2 in each of the first two cases; guessing the target and its peer costs $N\mu$ in the third. Adding the signature case gives the theorem's bound.

3.3 Subversion Security of iX3DH

Alice and Bob use the same reverse firewall. We describe Alice's in Fig. 2, then explain the subversions and disclosures covered by its security proof.

Alice's reverse firewall. The firewall intercepts Alice's raw public element $[x] \star E$, samples a fresh uniform mask $[\alpha]$, and sends $[x\alpha] \star E$ to the server. It returns $[\alpha\tau]$ to Alice when the server supplies $[\tau]$. Alice then uses the effective secret $[x\alpha\tau]$. The firewall checks that her next message contains the corresponding public element and a valid signature before forwarding it. The server may choose $[\tau]$ adaptively from the messages it has received. Explainability ensures that the client supplies the corresponding signature, whose value is fixed by uniqueness. These steps fit the publication phase of Fig. 1; offline storage and later delivery are unchanged.

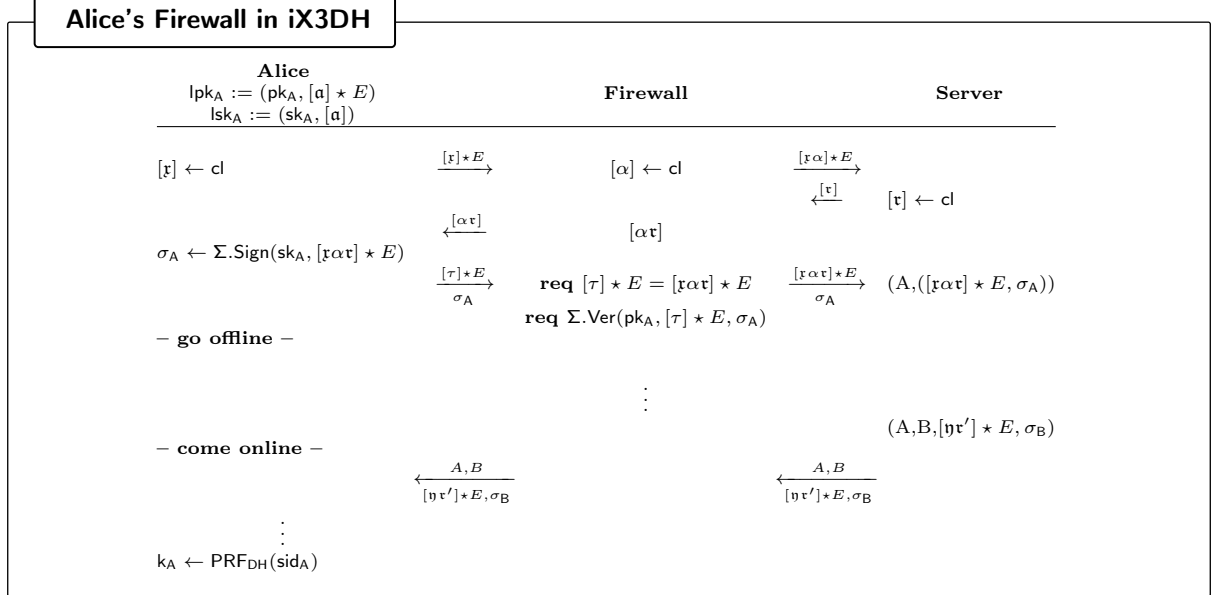


Fig. 2. Firewall W_A for Alice in iX3DH, showing honest sampling. A subverted Alice may bias $[x]$, and an adversarial server may choose $[\tau]$ adaptively. The firewall samples $[\alpha]$ independently and uniformly. These are ideal classes, as in Section 2.1.

Security analysis. We prove exfiltration resistance in the game of Definition 3. Key generation is honest, and subversions are functionality maintaining and explainable in the sense of Definition 1. Their random choices may be biased, secret-dependent or correlated across sessions. Explainability requires only that their messages and local results agree with an honest execution using some randomness.

Theorem 3. *The reverse firewall W_A in Fig. 2 maintains functionality for honest iX3DH. Moreover, with a perfectly correct unique signature scheme, it is exfiltration-resistant against functionality-maintaining, explainable implementations. Bob and the server may be controlled by a PPT adversary. Bob's firewall provides the same guarantees with Alice and the server controlled by a PPT adversary. In each case, the firewall protecting the party executes honestly.*

Proof. Functionality for honest implementations. Let Alice's raw secret be $[x]$ and the firewall mask be $[\alpha]$. The server sees $[x\alpha] \star E$ and returns $[\tau]$. Alice receives $[\alpha\tau]$ and consequently uses the effective secret $[x\alpha\tau]$. The signed public element checked by the firewall is precisely the public element for that secret. The same calculation applies at Bob's firewall. Both parties therefore form the same ordered sid and the same three DH shares, so they evaluate the KDF on identical inputs. Their final keys agree by the calculation in the proof of Theorem 1.

For a sequence of firewalls, replace $[\alpha]$ by the product of their masks. Commutativity gives the same calculation, so repeated honest firewall composition preserves correctness. Each honest endpoint erases the temporary values specified in Appendix B.

Exfiltration resistance. We prove the claim for functionality-maintaining, explainable implementations. Fix the honestly generated registered keys and all random-oracle functions, shared by both experiments. The adversarial party comprises Bob and the server. The game returns their state, the adversary's retained state, and the protocol transcript; local firewall traffic remains private.

Condition on the execution prefix before Alice sends her raw public element $X \in \mathcal{E}$. The subverted implementation may choose X as a function of its secrets and prior state. The firewall independently samples a uniform class $[\alpha]$ and sends $X' = [\alpha] \star X$. Regularity of the action makes X' uniform on $\mathcal{E}$, independently of the adversarial party's prior state. This is exactly the distribution of an honest initial public element $[x'] \star E$ for uniform $[x']$.

First compare the subverted, firewalled execution with an auxiliary execution using the honest implementation without a firewall. Fix the adversary's retained state as well. Couple the two executions to send the same X' and use the same coins for the adversarial party. It receives identical messages and reaches the same state in both worlds. In particular, its next randomizer $[\tau]$ may depend on X' and the whole preceding interaction; we do not sample it independently or fix it in advance. The second public element is $[\tau] \star X'$ in both worlds. By explainability, the subverted party's responses have an honest explanation on the same incoming messages. Every tape explaining X uses the same raw action class by regularity, so the local randomizer determines this second element. The client therefore supplies it with a valid signature, which uniqueness fixes to the honest signature on that message.

The remaining network messages and aborts are fixed by honest processing of the same inputs, as required by explainability. Invalid incoming messages follow the protocol's rejection rules in both executions. Induction over the invocations therefore gives the same transcript and adversarial state distribution in this auxiliary comparison. Apply the same calculation with the honest implementation in place of the subversion. It follows that an honest party with its firewall has the same transcript and adversarial state distribution as the auxiliary execution too. Averaging over the fixed keys and random functions, the two branches of LEAK have identical distributions, so the exfiltration advantage is zero. The argument also applies to Bob's firewall, conditioning on the incoming messages before each of Bob's outgoing steps.

Explainability supplies the honest response behavior, while uniqueness fixes the signature value. Together with the fresh mask, these properties prove the comparison in Definition 3. The AKE interfaces are treated in Appendix A.3 under Definition 19.

AKE security preservation. An AKE adversary also receives state and key disclosures and a Test answer. We must account for these interfaces to obtain a secrecy theorem. Definition 19 specifies the following extension of the base experiment.

After honest key generation, the adversary selects classical stateful replacement clients before any session or exposure query. They are functionality maintaining and explainable, and may use their private keys, bias their randomness and retain secrets across sessions. A state query to any session of either subverted participant in a tested exchange makes that test unfresh, before or after Test. Such a query returns the client's entire current memory. A long-term query returns only the fixed registered action and signing keys. Other clients remain exposable under the model, and honest clients retain the base session/source exposure rules. Sources now include all matching signed publications forwarded by an honest firewall, including those of subverted clients.

Explainability also determines the accepted key. Since the action is regular, the client's raw public element fixes its action class uniquely. Every honest tape explaining that element and the received randomizer therefore gives the same effective secret. The peer's public elements and the registered keys then determine the three DH shares, sid and the session key under the fixed KDF. The proof uses these common values; it does not need to find the explaining tape.

Corollary 1. *Assume the group action and cryptographic primitives of Theorem 2, and a perfectly correct unique signature scheme. Let $\mathcal{A}$ be a PPT adversary in Definition 19. Bound the total KDF queries of $\mathcal{A}$ and all replacement code by q_s , and put $N = \mu\ell$ and $Q = q_s + N + 1$. There are six QPT Q-GA-GDH reductions, three QPT PRF reductions with classical challenge queries, and one EU-CMA reduction with classical signing queries, with respective maximum advantages $\epsilon_g, \epsilon_p, \epsilon_{\text{sig}}$, such that*

$$\text{Adv}_{\Pi_W}^{\text{AKE-FS}}(\mathcal{A}) \leq (2N^2 + N\mu)(6Q\sqrt{\epsilon_g} + \epsilon_p) + \frac{1}{2}\mu\epsilon_{\text{sig}}.$$

Proof (sketch). Guess the target and the source or peer required by the base proof. Stop before a state query to either selected subverted client: a correctly guessed fresh execution cannot contain that query. Until this stop, replace those clients by simulators that sample the masked public elements directly. Fresh uniform firewall masks justify this replacement even when raw secrets depend on private keys and earlier oracle replies. Explainability and signature uniqueness then fix all observable messages, acceptance decisions and keys. Long-term queries return the same fixed keys; all other clients retain their actual state and query replies.

Embed the DH challenge in the selected masked public elements. In either mixed case, the owner of the exposed ephemeral state must be honest, so no subverted code needs an unknown long-term secret. The exact KDF oracle, both extraction bounds, and the PRF hop now apply as in Theorem 2. A source-free acceptance gives the remaining signature reduction. Section A.3 proves the replacement step and spells out each reduction, including private KDF queries made by other subverted clients.

Efficiency considerations. The firewall’s cost is two group-action evaluations per ephemeral publication: one to mask the initial public element and one to compute the expected final element after the server’s response. It also verifies the signature and multiplies classes to adjust the incoming randomizer. The network exchange consists of each party’s initial action element and signed final element, with the server supplying one represented class to each party and relaying their signed elements. The firewall processes these messages locally without an additional network round.

Key derivation uses one KDF evaluation on sid and the three DH values, followed by one PRF evaluation on sid . The session identifier consists of the ordered identities, long-term public keys and final ephemeral elements, all available locally when the parties derive the key.

The signature contribution depends on the instantiation. For IBIS, presented in Section 4, a signature contains the images of the deterministic N -torsion basis under the response isogeny. Its public key includes the curve, special basis and auxiliary isogeny data specified in Fig. 4. Concrete sizes and timings depend on the chosen group-action and IBIS parameters and encodings.

4 IBIS: an Isogeny-Based Unique Signature

In this section, we describe the first post-quantum unique digital signature scheme: IBIS³. It is derived from Leroux’s isogeny-based DeuringVUF based hash-and-sign signature scheme [34, section 5.2]. We discuss isogeny representations, describe the DeuringVUF based hash-and-sign signature scheme and lastly, we describe IBIS. We refer to the first two chapters of Leroux’s thesis [33] for background on quaternion algebras, supersingular isogenies and their interplay.

4.1 Isogeny Representation

Since isogenies are group homomorphisms, their most natural representation is their kernel. This representation is only efficient when the degree of the isogeny is smooth and the kernel generator is defined over a relatively small extension of $\mathbb{F}_{p^2}$ as one can use the Vélu formulas [43] or the square root Vélu formulas [9] to compute the isogeny. A composition of smooth degree isogenies can be represented by concatenating the representations of its factors. When the degree of the isogeny is not smooth, the kernel representation is no longer efficient.

The attacks on SIDH led to a new isogeny representation [39,40]. To represent an isogeny $\phi : E_1 \rightarrow E_2$ of generic degree a , one embeds ϕ into a higher dimensional smooth degree (say N) isogeny Φ in such a way that the evaluation of ϕ on torsion points can be retrieved from that of Φ on related points. This representation necessitates an *auxiliary isogeny* of degree $b = N - a$. Depending on whether such an auxiliary isogeny is available in dimension 1, 2 or 4, Φ will be a dimension 2, 4 or 8 isogeny respectively. We discuss the dimension 2 representation below and discuss the dimension 4 and 8 ones in Appendix C.

Dimension 2 representation. Let $\phi : E_1 \rightarrow E_2$ be an isogeny of degree a and let $N > a$ be a smooth integer coprime to d such that $E_1[N]$ is defined over a small extension of $\mathbb{F}_{p^2}$. Let $b = N - a$ and let $\psi : E_1 \rightarrow E_3$ be any isogeny of degree b . Then we have the following commutative diagram:

³ IBIS: Isogeny-Based unIque Signatures

$$\begin{array}{ccc}
E_1 & \xrightarrow{\phi} & E_2 \\
\psi \downarrow & & \downarrow \psi' \\
E_3 & \xrightarrow{\phi'} & E_4
\end{array}$$

The dimension 2 isogeny $\Theta : E_2 \times E_3 \rightarrow E_1 \times E_4$ given by $\Theta = \begin{pmatrix} \hat{\phi} & \hat{\psi} \\ -\psi' & \phi' \end{pmatrix}$, has kernel $\ker \Theta = \{(\phi(P), \psi(P)), P \in E_1[N]\}$ and degree $N = a + b$. Since Θ has smooth degree and accessible kernel, it can be evaluated on points $(P, 0) \in E_2 \times E_3$ to obtain $(\hat{\phi}(P), *)$, allowing to evaluate $\hat{\phi}$ efficiently. In practice, one chooses N to be a power of two. Note that the representation requires the knowledge of $\phi(E_1[N])$ and $\psi(E_1[N])$. If the person computing the isogeny representation knows the endomorphism ring of E_1 or b is smooth, then they can always generate this auxiliary isogeny ψ efficiently and use it to represent ϕ ; else they would need to use the dimension 4 or dimension 8 (see Appendix C) representation where the auxiliary isogeny can be efficiently and unconditionally computed.

To summarise, given an isogeny $\phi : E_1 \rightarrow E_2$ degree a and a smooth integer $N > a$ coprime to a such that $E_1[N]$ is defined over a small extension of $\mathbb{F}_{p^2}$, the data $\phi(E_1[N])$ is a representation of ϕ . To make sense of this representation, one embeds ϕ into a dimension 8 isogeny Θ by default, a dimension 4 isogeny Θ when $b = N - a$ can be written as the sum of two squares, or dimension 2 isogeny Θ when one has access to the any dimension 1 isogeny $\psi : E_1 \rightarrow E_3$ of degree b . The smaller the dimension of Θ , the more efficient the representation is. Schemes are hence designed in such a way that they can support the dimension 2 representation, or dimension 4 representation if the dimension 2 one is out of reach. In the rest of this section we will be using the dimension 2 representation. An important point to note here is that the auxiliary isogeny only depends on the degree of ϕ and is there only to enable the more efficient dimension 2 representation. Hence, if the degree of ϕ is known in advance, then the auxiliary isogeny ψ could be precomputed and made public in advance. This will be useful in the design of our unique signature scheme.

4.2 The DeuringVUF Based Hash-and-Sign Signature Scheme

Now that we know how to represent isogenies, let us briefly recall the DeuringVUF based hash-and-sign signature scheme from [34]. Our description is somewhat conceptual, as the technical details are arguably not much relevant as far as the follow-up construction is concerned. We refer the reader to [34] for further technical details.

In the previous section, we implicitly touched upon the fact that computing isogenies $\phi : E_1 \rightarrow E_2$ of generic degree q from their kernel is not efficient in general. In fact, this problem is believed to be hard when E_1 is a random supersingular elliptic curve and q is a large prime of cryptographic size. Nevertheless, thanks to the Deuring correspondence [20] and to refinements by Leroux [34], when the endomorphism ring $\text{End}(E_1)$ of E_1 is known, one can compute such an isogeny in polynomial time when the kernel generator K is sampled as a linear combination $K = [\alpha]R + [\beta]S$ of some purposely precomputed basis (R, S) of $E_1[q]$, which we call special basis. Here, since the party computing $\phi : E_1 \rightarrow E_2$ knows $\text{End}(E_1)$, they also generate an auxiliary isogeny $\psi : E_1 \rightarrow E_3$ of degree $N - q$ and returns $\phi(E_1[N])$ that represents ϕ (in dimension 8 by default) together with the auxiliary information $\psi(E_1[N])$ that enables a dimension 2 representation of ϕ . With this information, any third party can verify that the data $(\phi(E_1[N]), \psi(E_1[N]))$ represents the isogeny of kernel $\langle K \rangle$. We summarise the isogeny computation and the isogeny verification process with the following two algorithms, in which we assume that $N > q$ is fixed.

- $\text{data}(\phi) \leftarrow \text{ComputeIsogeny}_q(E, \text{End}(E), (R, S), x)$ which takes a supersingular curve E , its endomorphism ring $\text{End}(E)$, the special basis (R, S) of $E[q]$, and $x := (\alpha : \beta) \in \mathbb{P}^1(\mathbb{Z}/q\mathbb{Z})$, returns $\text{data}(\phi) = \phi(E[N])$ where the isogeny $\phi : E \rightarrow E_x$ has kernel $\langle K \rangle$ with $K = [\alpha]R + [\beta]S$.
- $\text{aux}(\phi) \leftarrow \text{ComputeAuxIsogeny}_q(E, \text{End}(E))$, which takes a supersingular curve E and its endomorphism ring $\text{End}(E)$ and returns $\text{aux}(\phi) = \psi(E[N])$ where $\psi : E \rightarrow E'$ is a random isogeny of degree $N - q$.
- $\text{accept/reject} \leftarrow \text{VerifyIsogeny}_q(E, (R, S), j_x, x, \text{data}(\phi), \text{aux}(\phi))$ which takes a supersingular curve E , the special basis (R, S) of $E[q]$, a j -invariant j_x , $x := (\alpha : \beta) \in \mathbb{P}^1(\mathbb{Z}/q\mathbb{Z})$, the data $\text{data}(\phi) = \phi(E[N])$

DeuringVUF

```

– Public Parameters –
A base prime  $p = 2^e \cdot f - 1$ 
A cryptographic size prime  $q, q \nmid (p - 1)$ 
An integer  $N = 2^{e'}$ ,  $e' = \lceil \log(q) \rceil \leq e$ 
 $(E_0, \text{End}(E_0))$  where  $E_0/\mathbb{F}_{p^2}$  is supersingular
The message space  $\mathcal{M} = \mathbb{P}^1((\mathbb{Z}/q\mathbb{Z}))$ 
 $\text{pp} = (p, E_0, \text{End}(E_0), q, N, \mathcal{M})$ 

– Key Generation –
Get the public parameters  $\text{pp}$ 
Sample a random isogeny  $\tau : E_0 \rightarrow E$ 
Recover  $\text{End}(E)$ 
Generate a special basis  $(R, S)$  of  $E[q]$ 
Set  $\text{pk} = (E, R, S)$ ,  $\text{sk} = \text{End}(E)$ 
return  $(\text{sk}, \text{pk})$ 

– Evaluation –
Get the input  $x \in \mathcal{M}$ ,  $\text{pp}$ ,  $\text{pk} = (E, R, S)$  and  $\text{sk} = \text{End}(E)$ 
 $\text{data}(\phi) \leftarrow \text{Computelsogeny}_q(E, \text{End}(E), (R, S), x)$ 
 $\text{aux}(\phi) \leftarrow \text{ComputeAuxIsogeny}_q(E, \text{End}(E))$ 
Recover the codomain  $E_x$  of  $\phi$ 
Set  $v = j(E_x)$  and  $\pi = (\text{data}(\phi), \text{aux}(\phi))$ 
return  $(v, \pi)$ 

– Verification –
Get the input  $x$ ,  $\text{pp}$ ,  $\text{pk} = (E, R, S)$ , the output  $v$  and the proof  $\pi$ 
Parse  $(\text{data}(\phi'), \text{aux}(\phi')) = \pi$  and  $j' = v$ 
return  $\text{VerifyIsogeny}_q(E, (R, S), j', x, \text{data}(\phi'), \text{aux}(\phi'))$ 

```

Fig. 3. The DeuringVUF from Leroux [34].

and the auxiliary data $\text{aux}(\phi) = \psi(E[N])$, and returns **accept** if $\text{data}(\phi)$ and $\text{aux}(\phi)$ form a valid dimension 2 representation of a q -isogeny $\phi : E \rightarrow E_x$ of kernel $\langle K \rangle$ with $K = [\alpha]R + [\beta]S$ and $j_x = j(E_x)$, or **reject** if not.

The DeuringVUF based hash-and-sign signature is derived from a Verifiable Unpredictable Function (VUF) with unconditional full uniqueness. This DeuringVUF is described in Fig. 3. The DeuringVUF is proven [34, Section 3.2] to be unpredictable under the assumption that Problem 1, which makes use of the q -FIXDIO oracle, is hard.

Definition 14. Given two odd primes $q \neq p$, a *FIXed Degree q -Isogeny Oracle* (q -FIXDIO) takes in input a supersingular elliptic curve E , and a point $K \in E[q]$ and outputs the j -invariant $j(E/\langle K \rangle)$ and an isogeny representation for the q -isogeny $\phi : E \rightarrow E/\langle K \rangle$.

Problem 1 (One-More Isogeny Problem (OMIP)). Given two odd primes $q \neq p$, let E be a random supersingular elliptic curve. Given access to the q -FIXDIO on input E , compute the j -invariant of the codomain of an isogeny of degree q not given as the output of the q -FIXDIO.

Remark 2. In [34], the algorithms Computelsogeny_q and ComputeAuxIsogeny are merged into one evaluation algorithm. In our description we separate them as we will move the ComputeAuxIsogeny step to key generation when designing our unique signature scheme.

The signature hash H_{sig} is a separate primitive from the AKE KDF H ; any random-oracle instantiation must domain-separate them. To turn the VUF into a hash-and-sign digital signature, one uses a hash function H_{sig} to hash the message m to be signed (concatenated with the public key) into the input

space of the VUF. The signing process consists in evaluating the VUF on $x = H_{\text{sig}}(\text{pk}||\text{m})$ and returning the output and the proof as the signature. Any third party can run the VUF verification algorithm on this output to accept or reject the signature. The unforgeability of the resulting signature protocol holds under the unpredictability of the VUF and the collision resistance of the hash function H_{sig} .

As mentioned earlier, the DeuringVUF satisfies unconditional full uniqueness, meaning that for a given a public key pk and a given input x , the output $v = j_x$ is unique. This is because j_x is the j -invariant of the co-domain of the isogeny $\phi : E \rightarrow E/\langle K \rangle$ where $K = [\alpha]R + [\beta]S$ with $x := (\alpha : \beta) \in \mathbb{P}^1(\mathbb{Z}/q\mathbb{Z})$. This means that for the signatures which are of the form (j_x, π) to be unique, we need to have unique proofs π . This will be our focus in the next section.

4.3 The IBIS Unique Signature Scheme

Our goal is to modify the construction in the previous section in such a way that given a public key pk , an input x and its corresponding output $v = j_x$, there exists a unique proof which is accepted by the verification algorithm. This will then lead to a hash-and-sign unique signature scheme.

The proof π is made of two quantities $\text{data}(\phi)$ and $\text{aux}(\phi)$. As discussed in Section 4.1, the auxiliary data $\text{aux}(\phi) = \psi(E[N])$ represents an auxiliary isogeny which depends only on E and on the degree q of the isogeny it is used to represent. Since q is part of the public parameters and E is generated during key generation, then this auxiliary data can be generated once and for all during key generation and appended to the public key. To highlight the fact that it does not depend on ϕ , we now denote it by $\text{aux}_q(E)$. We hence obtain $\text{pk} = (E, R, S, \text{aux}_q(E))$, and the proof π now contains $\text{data}(\phi)$ only. We are now left with establishing a verifiably unique way to represent $\text{data}(\phi) = \phi(E[N])$.

The data $\phi(E[N])$ is given by a basis (P, Q) of $E[N]$ together with its images $(\phi(P), \phi(Q))$ under ϕ . Any choice of (P, Q) would lead to a valid representation of the same information. To avoid these multiple representations arising from the choice of (P, Q) , we fix⁴ a deterministic basis (P_E, Q_E) of $E[N]$. The basis (P_E, Q_E) could be included in the public key. If the environment is memory constrained, an alternative would be to describe a deterministic algorithm that is used to generate (P_E, Q_E) from the curve E , we call this algorithm DetBasis_N . Having (P_E, Q_E) fixed once and for all, $\text{data}(\phi)$ is now reduced to $(\phi(P_E), \phi(Q_E))$.

While $(\phi(P_E), \phi(Q_E))$ encodes an isogeny $\phi : E \rightarrow E_x$ of kernel $\langle K \rangle = \langle [\alpha]R + [\beta]S \rangle$ with $x := (\alpha : \beta) \in \mathbb{P}^1(\mathbb{Z}/q\mathbb{Z})$, $(-\phi(P_E), -\phi(Q_E))$ encodes the isogeny $-\phi : E \rightarrow E_x$ which also has kernel $\langle K \rangle$. In fact, for every isomorphism $\eta : E_x \rightarrow E'_x$, $\eta \circ \phi$ is an isogeny of domain E with kernel $\langle K \rangle$, hence $(\eta \circ \phi(P_E), \eta \circ \phi(Q_E))$ would also pass the verification algorithm. In reality, the set of all isogenies ϕ' such that the data $(\phi'(P_E), \phi'(Q_E))$ passes the verification algorithm is the isomorphism class $\text{Valid}(\phi) = \{\eta \circ \phi, \eta \in \text{Iso}(E_x)\}$ of ϕ where $\text{Iso}(E_x) = \{\eta : E_x \rightarrow E'_x, j(E'_x) = j_x = j(E_x)\}$. This is a direct consequence of the fact that all valid proofs are representations of q -isogenies having the same kernel $\langle K \rangle = \langle [\alpha]R + [\beta]S \rangle$, implying that they are equal up-to post-composition by an isomorphism. To eliminate the multiple possibilities for the end curve E_x , we choose a canonical Montgomery curve F_x from the isomorphism class of E_x . Since this isomorphism class is determined by the unique output $v = j_x$, F_x is obtained as $F_x = \text{CanonicMontCurve}(j_x)$. This obliges the signer to post-compose the response ϕ with the correct isomorphism $\eta_x : E_x \rightarrow F_x$. We denote the resulting isogeny by $\phi_x = \eta_x \circ \phi$. While this eliminates most isomorphisms, it still does not prevent the signer from post-composing ϕ_x with an automorphism of F_x . In fact, we have restricted $\text{Valid}(\phi)$ to the "automorphism" class $\text{Valid}(\phi) = \{\eta \circ \phi_x, \eta \in \text{Aut}(F_x)\}$ of ϕ_x .

The idea here is to establish a deterministic and verifiable way to choose a unique element from $\text{Valid}(\phi)$. First let us note that $\text{Valid}(\phi)$ is a finite set. In fact, since we are in characteristic $p > 3$, then following [41, Theorem 10.1], the automorphism group of a given supersingular elliptic curve E' is

$$\text{Aut}(E') = \begin{cases} \{\iota, -\text{Id}, -\iota, \text{Id}\} & \text{if } j(E') = 1728; \\ \{\xi, \xi^2, -\text{Id}, -\xi, -\xi^2, \text{Id}\} & \text{if } j(E') = 0; \\ \{-\text{Id}, \text{Id}\} & \text{if } j(E') \neq 0, 1728; \end{cases} \quad (1)$$

where ι is an order 4 automorphism and ξ is an order 6 automorphism. To achieve our goal, we define a total strict order $<_\phi$ on $\text{Valid}(\phi)$. The output of the signature is then chosen to be the data corresponding

⁴ This is regular compression technique used isogeny-based cryptography and beyond.

to the smallest element $\phi_{\min}$ of $\text{Valid}(\phi)$. In the verification process, one additionally regenerates $\text{Valid}(\phi)$ and checks that the proof is the data corresponding the minimum $\phi_{\min}$ of $\text{Valid}(\phi)$.

Definition 15. Let $\phi : E \rightarrow F_x$ be an isogeny returned by VUF evaluation and let (P_E, Q_E) be a deterministically generated basis of $E[N]$, with $N > 4$. Let $\phi_1, \phi_2 \in \text{Valid}(\phi)$. We say that $\phi_1 <_\phi \phi_2$ if one of the following holds:

- (i) $x_{\phi_1(P_E)} <_2 x_{\phi_2(P_E)}$,
- (ii) $x_{\phi_1(P_E)} = x_{\phi_2(P_E)}$ and $y_{\phi_1(P_E)} <_2 y_{\phi_2(P_E)}$,

where $<_2$ is a total strict order on $\mathbb{F}_{p^2}$.

Theorem 4. The binary relation $<_\phi$ on $\text{Valid}(\phi)$ is a total strict order.

The proof of Theorem 4 is deferred to Appendix A.4.

Having constructed all the required tools, we are now ready to describe our unique signature scheme IBIS. This is done in Fig. 4. Note that the output $v = j_x$ is not explicitly apparent in the signature, this is because it can be recovered from $\text{data}(\phi)$ and, as a consequence of unconditional full uniqueness of the VUF, the verification only succeeds if the recovered value $v = j_x$ is the correct one.

Since IBIS is obtained by moving auxiliary data (whose role is to enable an efficient dimension 2 representation of the signature isogeny ϕ) from the signature to the public key, and establishing a verifiably unique way to represent the response isogeny ϕ , then the existential unforgeability under chosen messages attacks (EU-CMA) of IBIS also holds under the unpredictability of the VUF and the collision resistance of the hash function H_{sig} . We now formally prove that IBIS signatures are unique.

Theorem 5. The IBIS signature scheme has unique signatures.

Proof. Let σ_1 and σ_2 be two valid IBIS signatures for the same message m under a public key $\text{pk} = (E, R, S, \text{aux}_q(E))$. Let (P_E, Q_E) be the deterministic basis of $E[N]$. Write $\sigma_i = \text{data}(\phi_i) = (\phi_i(P_E), \phi_i(Q_E))$. Then both ϕ_1 and ϕ_2 are q -isogenies with the same kernel $\langle K \rangle$ where $K = [\alpha]R + [\beta]S$ with $(\alpha : \beta) = H_{\text{sig}}(\text{pk}||m)$, meaning that they are equal up to post-composition by an isomorphism η and their codomains have the same j -invariant j_x . Since both signatures are valid, then by design of the verification algorithm, ϕ_1 and ϕ_2 have the same codomain $F_x = \text{CanonicMontCurve}(j_x)$, implying that η is in fact an automorphism of F_x . This leads to $\text{Valid}(\phi_1) = \text{Valid}(\phi_2)$, further implying that $\text{minValid}(\phi_1) = \text{minValid}(\phi_2)$. Since σ_1 and σ_2 are valid, then $\phi_1 = \text{minValid}(\phi_1)$ and $\phi_2 = \text{minValid}(\phi_2)$, implying that $\phi_1 = \phi_2$. We have $\sigma_1 = (\phi_1(P_E), \phi_1(Q_E)) = (\phi_2(P_E), \phi_2(Q_E)) = \sigma_2$. $\square$

Together with EU-CMA security, signature uniqueness gives IBIS strong unforgeability under chosen-message attacks (SUF-CMA). As stated by the NIST in its latest report [3] announcing the third round candidates for the post-quantum additional signatures standardisation process, this property is desirable. When, compared to other quaternion isogeny-based signatures such as SQIsign [1], PRISM [7], Water-SQI/PRISMO [27] and the DeuringVUF based signature, IBIS is the only one which is SUF-CMA by design. Recently, a strongly-unforgeable variant of PRISM was proposed [8], but the strong-unforgeability property is achieved under an assumption which is stronger than the one needed for EUF-CMA in PRISM.

IBIS Unique signature scheme

– **Public Parameters** –

Get $p, E_0, \text{End}(E_0), q, N$ as in the DeuringVUF
 A collision resistant hash function $H_{\text{sig}} : \{0, 1\}^* \rightarrow \mathbb{P}^1(\mathbb{Z}/q\mathbb{Z})$
 $\text{pp} = (p, E_0, \text{End}(E_0), q, N, H_{\text{sig}})$

– **Key Generation** –

Get the public parameters pp
 Sample a random isogeny $\tau : E_0 \rightarrow E$
 Recover $\text{End}(E)$
 Generate a special basis (R, S) of $E[q]$
 $\text{aux}_q(E) \leftarrow \text{ComputeAuxIsogeny}_q(E, \text{End}(E))$
 Set $\text{pk} = (E, R, S, \text{aux}_q(E))$, $\text{sk} = \text{End}(E)$
return (sk, pk)

– **Signing** –

Get the message m , $\text{pp}, \text{pk} = (E, R, S, \text{aux}_q(E))$, $\text{sk} = \text{End}(E)$
 Set $x \leftarrow H_{\text{sig}}(\text{pk}||m)$
 $\text{data}(\phi) \leftarrow \text{ComputeIsogeny}_q(E, \text{End}(E), (R, S), x)$ where $\phi : E \rightarrow E_x$
 Recover $\text{data}(\phi')$ where $\phi' = \eta_x \circ \phi : E \rightarrow F_x = \text{CanonicMontCurve}(j_x)$
 Compute $\text{Valid}(\phi')$ and determine $\phi_{\min} = \text{minValid}(\phi')$
 Recover $\text{data}(\phi_{\min})$
return $\sigma = \text{data}(\phi_{\min})$

– **Verification** –

Get $\text{pp}, \text{pk} = (E, R, S, \text{aux}_q(E))$, m and σ
 Set $(P_E, Q_E) \leftarrow \text{DetBasis}_N(E)$
 Parse $\text{data}(\phi') = \sigma$ as $(\phi'(P_E), \phi'(Q_E))$
 Extract the codomain E' of ϕ'
if $E' \neq \text{CanonicMontCurve}(j(E'))$ **then**
 return reject
end if
 Set $x \leftarrow H_{\text{sig}}(\text{pk}||m)$
if $\text{VerifyIsogeny}_q(E, (R, S), j(E'), x, \text{data}(\phi'), \text{aux}_q(E)) \neq \text{accept}$ **then**
 return reject
end if
 Compute $\text{Valid}(\phi')$ and determine $\phi_{\min} = \text{minValid}(\phi')$
if $\phi'(P_E) = \phi_{\min}(P_E)$ **then**
 return accept
end if
return reject

Fig. 4. IBIS unique digital signature scheme.

References

1. Aardal, M.A., Adj, G., Aranha, D.F., Basso, A., Canales Martínez, I.A., Chávez-Saab, J., Corte-Real Santos, M., Dartois, P., De Feo, L., Duparc, M., Eriksen, J.K., Fouotsa, T.B., Gazzoni Filho, D.L., Hess, B., Kohel, D., Leroux, A., Longa, P., Maino, L., Meyer, M., Nakagawa, K., Onuki, H., Panny, L., Patranabis, S., Petit, C., Pope, G., Reijnders, K., Robert, D., Rodríguez-Henríquez, F., Schaeffler, S., Wesolowski, B.: SQIsign. Tech. rep., National Institute of Standards and Technology (2025), <https://sqisign.org>
2. Abdalla, M., Eisenhofer, T., Kiltz, E., Kunzweiler, S., Riepel, D.: Password-authenticated key exchange from group actions. In: Annual international cryptology conference. pp. 699–728. Springer (2022)
3. Alagic, G., Bros, M., Ciadoux, P., Dang, Q., Dang, T.H., Kelsey, J., Lichtinger, J., Liu, Y.K., Miller, C., Moody, D., Peralta, R., Perlner, R., Robinson, A., Silberg, H., Smith-Tone, D., Waller, N.: Status Report on the Second Round of the Additional Digital Signature Schemes for the NIST Post-Quantum Cryptography Standardization Process. (National Institute of Standards and Technology, Gaithersburg, MD) NIST Internal Report (IR) NIST IR 8610 (2026), <https://doi.org/10.6028/NIST.IR.8610>

4. Ambainis, A., Hamburg, M., Unruh, D.: Quantum security proofs using semi-classical oracles. In: Boldyreva, A., Micciancio, D. (eds.) *Advances in Cryptology – CRYPTO 2019, Part II. Lecture Notes in Computer Science*, vol. 11693, pp. 269–295. Springer, Cham, Switzerland, Santa Barbara, CA, USA (Aug 18–22, 2019). https://doi.org/10.1007/978-3-030-26951-7_10
5. Ateniese, G., Magri, B., Venturi, D.: Subversion-resilient signature schemes. In: Ray, I., Li, N., Kruegel, C. (eds.) *ACM CCS 2015: 22nd Conference on Computer and Communications Security*. pp. 364–375. ACM Press, Denver, CO, USA (Oct 12–16, 2015). <https://doi.org/10.1145/2810103.2813635>
6. Banegas, G., Bernstein, D.J., Campos, F., Chou, T., Lange, T., Meyer, M., Smith, B., Sotáková, J.: CTIDH: faster constant-time CSIDH. *IACR Transactions on Cryptographic Hardware and Embedded Systems* **2021**(4), 351–387 (2021). <https://doi.org/10.46586/tches.v2021.i4.351-387>, <https://tches.iacr.org/index.php/TCHES/article/view/9069>
7. Basso, A., Borin, G., Castryck, W., Corte-Real Santos, M., Invernizzi, R., Leroux, A., Maino, L., Vercauteren, F., Wesolowski, B.: PRISM: Simple and Compact Identification and Signatures from Large Prime Degree Isogenies. In: Jager, T., Pan, J. (eds.) *Public-Key Cryptography – PKC 2025*. pp. 300–332. Springer Nature Switzerland, Cham (2025)
8. Basso, A., Borin, G., Castryck, W., Santos, M.C.R., Invernizzi, R., Leroux, A., Maino, L., Vercauteren, F., Wesolowski, B.: PRISM with a pinch of salt: Simple, efficient and strongly unforgeable signatures from isogenies. *Cryptology ePrint Archive*, Paper 2026/443 (2026), <https://eprint.iacr.org/2026/443>
9. Bernstein, D.J., De Feo, L., Leroux, A., Smith, B.: Faster computation of isogenies of large prime degree. *Open Book Series* **4**(1), 39–55 (2020)
10. Beullens, W., Kleinjung, T., Vercauteren, F.: CSI-FiSh: Efficient isogeny based signatures through class group computations. In: Galbraith, S.D., Moriai, S. (eds.) *Advances in Cryptology – ASIACRYPT 2019, Part I. Lecture Notes in Computer Science*, vol. 11921, pp. 227–247. Springer, Cham, Switzerland, Kobe, Japan (Dec 8–12, 2019). https://doi.org/10.1007/978-3-030-34578-5_9
11. Bhargavan, K., Jacomme, C., Kiefer, F., Schmidt, R.: Formal verification of the PQXDH post-quantum key agreement protocol for end-to-end secure messaging. In: Balzarotti, D., Xu, W. (eds.) *USENIX Security 2024: 33rd USENIX Security Symposium*. USENIX Association, Philadelphia, PA, USA (Aug 14–16, 2024), <https://www.usenix.org/conference/usenixsecurity24/presentation/bhargavan>
12. Bonnetain, X., Schrottenloher, A.: Quantum security analysis of CSIDH. In: Canteaut, A., Ishai, Y. (eds.) *Advances in Cryptology – EUROCRYPT 2020, Part II. Lecture Notes in Computer Science*, vol. 12106, pp. 493–522. Springer, Cham, Switzerland, Zagreb, Croatia (May 10–14, 2020). https://doi.org/10.1007/978-3-030-45724-2_17
13. Brendel, J., Fischlin, M., Günther, F., Janson, C., Stebila, D.: Towards post-quantum security for Signal’s X3DH handshake. In: Dunkelman, O., Jacobson, Jr., M.J., O’Flynn, C. (eds.) *SAC 2020: 27th Annual International Workshop on Selected Areas in Cryptography. Lecture Notes in Computer Science*, vol. 12804, pp. 404–430. Springer, Cham, Switzerland, Halifax, NS, Canada (Virtual Event) (Oct 21–23, 2020). https://doi.org/10.1007/978-3-030-81652-0_16
14. Castryck, W., Decru, T.: An efficient key recovery attack on SIDH. In: Hazay, C., Stam, M. (eds.) *Advances in Cryptology – EUROCRYPT 2023, Part V. Lecture Notes in Computer Science*, vol. 14008, pp. 423–447. Springer, Cham, Switzerland, Lyon, France (Apr 23–27, 2023). https://doi.org/10.1007/978-3-031-30589-4_15
15. Castryck, W., Lange, T., Martindale, C., Panny, L., Renes, J.: CSIDH: An efficient post-quantum commutative group action. In: Peyrin, T., Galbraith, S. (eds.) *Advances in Cryptology – ASIACRYPT 2018, Part III. Lecture Notes in Computer Science*, vol. 11274, pp. 395–427. Springer, Cham, Switzerland, Brisbane, Queensland, Australia (Dec 2–6, 2018). https://doi.org/10.1007/978-3-030-03332-3_15
16. Chakraborty, S., Dziembowski, S., Nielsen, J.B.: Reverse firewalls for actively secure MPCs. In: Micciancio, D., Ristenpart, T. (eds.) *Advances in Cryptology – CRYPTO 2020, Part II. Lecture Notes in Computer Science*, vol. 12171, pp. 732–762. Springer, Cham, Switzerland, Santa Barbara, CA, USA (Aug 17–21, 2020). https://doi.org/10.1007/978-3-030-56880-1_26
17. Chakraborty, S., Magliocco, L., Magri, B., Venturi, D.: Key exchange in the post-snowden era: Universally composable subversion-resilient PAKE. In: Chung, K.M., Sasaki, Y. (eds.) *Advances in Cryptology – ASIACRYPT 2024, Part V. Lecture Notes in Computer Science*, vol. 15488, pp. 101–133. Springer, Singapore, Singapore, Kolkata, India (Dec 9–13, 2024). https://doi.org/10.1007/978-981-96-0935-2_4
18. Dartois, P., Eriksen, J.K., Fouotsa, T.B., Merdy, A.H.L., Invernizzi, R., Robert, D., Rueger, R., Vercauteren, F., Wesolowski, B.: PEGASIS: Practical effective class group action using 4-dimensional isogenies. In: Kalai, Y.T., Kamara, S.F. (eds.) *Advances in Cryptology – CRYPTO 2025, Part I. Lecture Notes in Computer Science*, vol. 16000, pp. 67–99. Springer, Cham, Switzerland, Santa Barbara, CA, USA (August 17–21, 2025). https://doi.org/10.1007/978-3-032-01855-7_3
19. Dartois, P., Eriksen, J.K., Invernizzi, R., Vercauteren, F.: qt-pegasis: Simpler and faster effective class group actions. *Cryptology ePrint Archive*, Report 2025/1859 (2025), <https://eprint.iacr.org/2025/1859>

20. Deuring, M.: Die typen der multiplikatorenringe elliptischer funktionenkörper: G. herglotz zum 60. geburtstag gewidmet. In: *Abhandlungen aus dem mathematischen Seminar der Universität Hamburg*, vol. 14, pp. 197–272. Springer (1941)
21. Dobson, S., Galbraith, S.D.: Post-quantum Signal key agreement from SIDH. In: Cheon, J.H., Johansson, T. (eds.) *Post-Quantum Cryptography - 13th International Workshop, PQCrypto 2022*, pp. 422–450. Springer, Cham, Switzerland, Virtual Event (Sep 28–30, 2022). https://doi.org/10.1007/978-3-031-17234-2_20
22. Dodis, Y., Magri, B., Stephens-Davidowitz, N., Tselekounis, Y.: Guarding the signal: Secure messaging with reverse firewalls. In: Kalai, Y.T., Kamara, S.F. (eds.) *Advances in Cryptology - CRYPTO 2025 - 45th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 17–21, 2025, Proceedings, Part VIII. Lecture Notes in Computer Science*, vol. 16007, pp. 39–74. Springer (2025). https://doi.org/10.1007/978-3-032-01913-4_2, https://doi.org/10.1007/978-3-032-01913-4_2
23. Dodis, Y., Mironov, I., Stephens-Davidowitz, N.: Message transmission with reverse firewalls—secure communication on corrupted machines. In: Robshaw, M., Katz, J. (eds.) *Advances in Cryptology – CRYPTO 2016, Part I. Lecture Notes in Computer Science*, vol. 9814, pp. 341–372. Springer Berlin Heidelberg, Germany, Santa Barbara, CA, USA (Aug 14–18, 2016). https://doi.org/10.1007/978-3-662-53018-4_13
24. Don, J., Fehr, S., Majenz, C.: The measure-and-reprogram technique 2.0: Multi-round fiat-shamir and more. In: Micciancio, D., Ristenpart, T. (eds.) *Advances in Cryptology – CRYPTO 2020, Part III. Lecture Notes in Computer Science*, vol. 12172, pp. 602–631. Springer, Cham, Switzerland, Santa Barbara, CA, USA (Aug 17–21, 2020). https://doi.org/10.1007/978-3-030-56877-1_21
25. Duman, J., Hartmann, D., Kiltz, E., Kunzweiler, S., Lehmann, J., Riepel, D.: Group action key encapsulation and non-interactive key exchange in the QROM. In: Agrawal, S., Lin, D. (eds.) *Advances in Cryptology – ASIACRYPT 2022, Part II. Lecture Notes in Computer Science*, vol. 13792, pp. 36–66. Springer, Cham, Switzerland, Taipei, Taiwan (Dec 5–9, 2022). https://doi.org/10.1007/978-3-031-22966-4_2
26. Duverger, K., Fouque, P.A., Jacomme, C., Niot, G., Onete, C.: Subversion-resilient key-exchange in the post-quantum world. In: *CCS 2025 - 32nd ACM Conference on Computer and Communications Security*. Taipei, Taiwan (October 2025), <https://inria.hal.science/hal-05242187>
27. Fouotsa, T.B.: WaterSQI and PRISMO: Quaternion Signatures for Supersingular Isogeny Group Actions. In: Daemen, J., Thomé, E. (eds.) *Advances in Cryptology – EUROCRYPT 2026*, pp. 573–603. Springer Nature Switzerland, Cham (2026)
28. Ganesh, C., Magri, B., Venturi, D.: Cryptographic reverse firewalls for interactive proof systems. In: Czumaj, A., Dawar, A., Merelli, E. (eds.) *ICALP 2020: 47th International Colloquium on Automata, Languages and Programming. Leibniz International Proceedings in Informatics (LIPIcs)*, vol. 168, pp. 55:1–55:16. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, Saarbrücken, Germany (Jul 8–11, 2020). <https://doi.org/10.4230/LIPIcs.ICALP.2020.55>
29. Hashimoto, K., Katsumata, S., Kwiatkowski, K., Prest, T.: An efficient and generic construction for Signal’s handshake (X3DH): Post-quantum, state leakage secure, and deniable. *Journal of Cryptology* **35**(3), 17 (Jul 2022). <https://doi.org/10.1007/s00145-022-09427-1>
30. Houben, M.: Deterministic algorithms for class group actions. In: Kalai, Y.T., Kamara, S.F. (eds.) *Advances in Cryptology – CRYPTO 2025, Part I. Lecture Notes in Computer Science*, vol. 16000, pp. 100–130. Springer, Cham, Switzerland, Santa Barbara, CA, USA (August 17–21, 2025). https://doi.org/10.1007/978-3-032-01855-7_4
31. Jaeger, J.: Nonadaptive one-way to hiding implies adaptive quantum reprogramming. *Cryptology ePrint Archive, Report 2024/797* (2024), <https://eprint.iacr.org/2024/797>
32. Kret, E., Schmidt, R.: The pqxdh key agreement protocol. url: <https://signal.org/docs/specifications/pqxdh/pqxdh.pdf> (2023)
33. Leroux, A.: Quaternion algebra and isogeny-based cryptography. Theses, Ecole doctorale de l’Institut Polytechnique de Paris (2022)
34. Leroux, A.: Verifiable random function from the deuring correspondence and higher dimensional isogenies. In: Fehr, S., Fouque, P.A. (eds.) *Advances in Cryptology – EUROCRYPT 2025, Part VII. Lecture Notes in Computer Science*, vol. 15607, pp. 167–194. Springer, Cham, Switzerland, Madrid, Spain (May 4–8, 2025). https://doi.org/10.1007/978-3-031-91098-2_7
35. Mironov, I., Stephens-Davidowitz, N.: Cryptographic reverse firewalls. In: Oswald, E., Fischlin, M. (eds.) *Advances in Cryptology – EUROCRYPT 2015, Part II. Lecture Notes in Computer Science*, vol. 9057, pp. 657–686. Springer Berlin Heidelberg, Germany, Sofia, Bulgaria (Apr 26–30, 2015). https://doi.org/10.1007/978-3-662-46803-6_22
36. Page, A., Robert, D.: Introducing clapoti(s): Evaluating the isogeny class group action in polynomial time. *Cryptology ePrint Archive, Report 2023/1766* (2023), <https://eprint.iacr.org/2023/1766>
37. Panny, L., Petit, C., Stopar, M.: KLaPoTi: An asymptotically efficient isogeny group action from 2-dimensional isogenies. *Cryptology ePrint Archive, Report 2024/1844* (2024), <https://eprint.iacr.org/2024/1844>

38. Peikert, C.: He gives C-sieves on the CSIDH. In: Canteaut, A., Ishai, Y. (eds.) Advances in Cryptology – EUROCRYPT 2020, Part II. Lecture Notes in Computer Science, vol. 12106, pp. 463–492. Springer, Cham, Switzerland, Zagreb, Croatia (May 10–14, 2020). https://doi.org/10.1007/978-3-030-45724-2_16
39. Robert, D.: Evaluating isogenies in polylogarithmic time. Cryptology ePrint Archive, Paper 2022/1068 (2022), <https://eprint.iacr.org/2022/1068>
40. Robert, D.: On the efficient representation of isogenies: a survey for nutmic 2024. In: International Conference on Number-Theoretic Methods in Cryptology. pp. 3–84. Springer (2024)
41. Silverman, J.H.: The Arithmetic of Elliptic Curves, Graduate Texts in Mathematics, vol. 106. Springer, New York, 2nd edn. (2009). <https://doi.org/10.1007/978-0-387-09494-6>
42. Urbanik, D., Jao, D.: SoK: The Problem Landscape of SIDH. In: Proceedings of the 5th ACM on ASIA Public-Key Cryptography Workshop. p. 53–60. APKC '18, Association for Computing Machinery, New York, NY, USA (2018). <https://doi.org/10.1145/3197507.3197516>, <https://doi.org/10.1145/3197507.3197516>
43. Vélú, J.: Isogénies entre courbes elliptiques. Comptes-Rendus de l’Académie des Sciences **273**, 238–241 (1971)
44. Zhandry, M.: How to construct quantum random functions. In: 53rd Annual Symposium on Foundations of Computer Science. pp. 679–687. IEEE Computer Society Press, New Brunswick, NJ, USA (Oct 20–23, 2012). <https://doi.org/10.1109/FOCS.2012.37>

Declaration on the Use of AI Tools

During the preparation of this manuscript, the authors employed AI tools to assist with language editing and with some of the technical investigation. The authors assume full responsibility for the technical correctness, veracity, and theoretical integrity of the presented material.

A Supplementary Proofs

A.1 Proof of Theorem 1

Proof. In an honest execution, Alice and Bob receive the same public keys and final ephemeral elements, so they assign the same session identifier. Their roles and peer identities also agree with the partnering definition. The group action is commutative, so both compute the shares $[\mathbf{rrb}] \star E$, $[\mathbf{ayr}'] \star E$, and $[\mathbf{rryr}'] \star E$. They therefore obtain the same KDF output and apply the PRF to the same session identifier. Perfect correctness of the signature scheme ensures that both signature checks succeed. Thus both parties accept with the same session key.

A.2 Proof of Theorem 2

Proof. We use the source-based freshness model in Appendix B. The QROM argument below combines an exact KDF simulation with a query-extraction bound before Test and ordinary O2H after Test. Let $\mathcal{A}$ be a QPT adversary playing $G_{\Pi_{\text{AKE}}}^{\text{FS}}(\mu, \ell)$ with challenger $\mathcal{C}$, with advantage $\text{Adv}_{\Pi_{\text{AKE}}}^{\text{AKE-FS}}(\mathcal{A})$. We organise fresh executions by the sources of the accepted peer share and their exposures, as in Table 1.

In the security game, protocol messages are those delivered to Alice and Bob. The adversary also controls the server and chooses the randomizers applied to their ephemeral keys.

In Case-III, both ephemeral secrets are revealed. The adversary can then compute DH_1 , DH_2 , and DH_3 from these secrets and the public keys. The freshness conditions in Appendix B exclude this case.

We assume without loss of generality that $\mathcal{A}$ makes a valid Test query; otherwise its advantage is zero. Write $\Pr[G_j]$ for its probability of guessing b correctly in game G_j , and let $\text{Adv}_{G_j}^{\mathcal{A}} := |\Pr[G_j] - \frac{1}{2}|$. All cases start with the following two games.

Whenever a proof game aborts, it replaces the adversary’s final guess with an independent uniform bit. An aborted execution therefore has winning probability $1/2$. A guess is rejected as soon as it is known to be wrong; the absence of a completed partner at an intermediate stage is not by itself grounds for aborting.

Game G_0 . This game is identical to the original security game. We thus have

$$\text{Adv}_{G_0}^{\mathcal{A}} = \text{Adv}_{\Pi_{\text{AKE}}}^{\text{AKE-FS}}(\mathcal{A}).$$

Table 1. Exposure cases for an initiating or responding test session.

Case	Peer-share sources	State revealed		Long-term key kept secret
		Target	Any source	
I	Present	No	No	None required
II,t	Present	Yes	No	Target owner's
II,s	Present	No	Yes	Peer's
IV	Absent	Either	—	Peer until target acceptance
III	Present	Yes	Yes	Excluded by freshness

“Any source” means at least one session in $\text{Src}(T)$; a source may be pending or have a different sid . All freshness conditions still apply. In Case IV, revealing the target state also requires the target owner's long-term key to remain secret.

Game G_1 . This game is identical to G_0 , except that it aborts if two accepted oracles with the same defined sid have different session keys. This includes two oracles owned by the same party. Equal session identifiers fix the same ordered identities, long-term public keys, and final ephemeral public elements. Writing the latter as $[u] \star E$ and $[v] \star E$, both oracles compute $[ub] \star E$, $[av] \star E$, and $[uv] \star E$ by commutativity of the group action. They evaluate the KDF on the same sid and triple, and the PRF on that same sid . Their keys therefore agree, even when the messages were scheduled by the adversary. Hence $\Pr[G_0] = \Pr[G_1]$.

Partition of fresh executions. Let T be the tested session. Write $e_T = 1$ if its state was revealed and $e_S = 1$ if the state of any session in $\text{Src}(T)$ was revealed. First assign executions with $\text{Src}(T) = \emptyset$ to C_{IV} . For the remaining executions, define C_I by $(e_T, e_S) = (0, 0)$, $C_{\text{II,t}}$ by $(e_T, e_S) = (1, 0)$, and $C_{\text{II,s}}$ by $(e_T, e_S) = (0, 1)$. The pair $(1, 1)$ is excluded by freshness. All these events include final freshness, and they cover both tested roles. In particular, an unpartnered session with an honest source belongs to a DH case, not to C_{IV} .

Let W denote a correct final guess. For each fresh case C , define its signed contribution by

$$\delta_C := \Pr[W \wedge C] - \frac{1}{2} \Pr[C].$$

Unfresh executions end with a fair guess, so

$$\text{Adv}_{G_1}^A = \left| \sum_C \delta_C \right| \leq |\delta_I| + |\delta_{\text{II,t}}| + |\delta_{\text{II,s}}| + |\delta_{\text{IV}}|.$$

This is an identity about execution events, not permission to ignore queries outside a chosen case. A reduction embedding an unknown secret must handle its reveal or stop with a justified abort rule before continuing.

Choosing a source. When $\text{Src}(T)$ is nonempty, let $S(T)$ be its earliest published source. This choice is fixed at target acceptance. A valid Test has a peer distinct from its owner, and every source belongs to that peer. Hence $S(T) \neq T$, even when $S(T)$ was itself started as a self session. The two raw ephemeral secrets used in Case I therefore occupy distinct sessions and were sampled independently.

An exact simulation of the KDF. Write A, B for the ordered long-term action public keys and X, Y for the ordered final ephemeral elements. Their session identifier is $c = (I, J, A, B, X, Y)$, where I, J are the ordered initiator and responder identities. For $U = [u] \star E$ and $V = [v] \star E$, put $\text{DH}_E(U, V) = [uv] \star E$. Regularity makes this value unique. Let $D_E(U, V, W)$ be its decision predicate from Definition 13. For a triple $w = (w_1, w_2, w_3)$, define

$$\text{Valid}(c, w) = D_E(X, B, w_1) \wedge D_E(A, Y, w_2) \wedge D_E(X, Y, w_3).$$

Sample independent random functions R on session identifiers and $\mathcal{J}$ on full KDF inputs, and answer KDF queries with

$$F(c, w) = \begin{cases} R(c), & \text{Valid}(c, w) = 1, \\ \mathcal{J}(c, w), & \text{Valid}(c, w) = 0. \end{cases} \quad (2)$$

Malformed inputs use $\mathcal{J}$. Every well-formed c has exactly one valid triple $w(c)$. Distinct valid inputs therefore use distinct, independent R entries; invalid inputs use independent $\mathcal{J}$ entries. Thus F is exactly a random function. An honest session with $\text{sid} = c$ computes $w(c)$, so its seed is exactly $R(c)$. This equality holds from the start, including before the adversary delivers the randomizers defining X, Y . It also covers unrelated source exchanges and repeated session identifiers.

The circuit computes the three decision bits, uses a controlled call to R or $\mathcal{J}$, and uncomputes the bits. It uses at most six coherent decision queries per KDF query. The call to R is active only on the valid branch. For a polynomial query bound, sufficiently independent functions implement $R, \mathcal{J}$ efficiently, with the same bounded-query behavior as random functions [44]. All calls made by the simulation and extractors count towards that bound. This construction uses the public-context reindexing idea of [25, Section 4.1]; here the decision arguments are unrestricted.

In this simulation, acceptance stores the sid and a symbolic key. The simulator evaluates its seed only when a key is returned by `RevSessKey` or `Test`, and caches evaluations. This has the same observable distribution as immediate evaluation: no protocol message depends on the key, and the state returned by `RevState` excludes erased DH shares and seeds. A key-reveal flag is recorded before any oracle call answering that reveal. At most $N = \mu\ell$ session seeds are evaluated internally. We use $Q = q_s + N + 1 \geq 1$ as a common bound for the oracle calls in either phase.

Extracting a query before the transcript is fixed. We give the needed query bound explicitly. It combines the telescoping argument of [24] with the Fourier representation of a random oracle used in [31]. It does not assume that measuring a query preserves the adversary's original continuation.

The difficulty is the order of information. A query can contain the right DH triple before the reduction knows the randomizers needed to decode it. We therefore extract a query together with a matching later transcript. Setting one label's answers to zero in the continuation lets us bound the probability of obtaining both. This modification is part of the extractor, not a change to the protocol or its real oracle.

Consider any algorithm making at most Q controlled calls to R , with an initial state independent of R . Its final classical output is (c, z) . Fix a predicate on that output and let P_c project onto permitted outputs with label c . Intermediate classical interactions are purified with their records retained. In the Fourier representation, the oracle table has cells $D[c]$ initially zero. An active query with label c and Fourier answer register y replaces $D[c]$ by $D[c] \oplus y$. Only oracle calls touch this table. Let L_c project onto $D[c] \neq 0$. For the final state $|\psi\rangle$, set

$$p = \sum_c \|P_c L_c |\psi\rangle\|^2.$$

The label c can be chosen adaptively. A zero cell need not mean that it was never queried: successive queries can cancel in the Fourier representation. We use only that a run answering zero at c leaves this cell zero, and that swapping it with a fresh zero cell has no effect on the zero-cell subspace.

There is an extractor with success probability p_{ext} such that

$$p \leq 4Q^2 p_{\text{ext}}. \quad (3)$$

It chooses a uniform call index and a fair bit, runs the original prefix, and measures the call's activity and label. An inactive call fails. For the measured label c , the chosen call is answered normally or with zero according to the bit; every later call at c is answered with zero. Other labels keep their original answers. The extractor succeeds exactly when the final output satisfies P_c , and retains z .

To prove the bound, fix c . Write O for the normal call, O_c for the call with value zero at c , and T_c for the projection onto an active query at c . Then $O - O_c = (O - O_c)T_c$. Let $|\psi_j\rangle$ be the state before call j in the original run, and $V_{c,j}$ the suffix after that call, using O_c throughout. The run using only O_c leaves $D[c] = 0$, so telescoping gives

$$P_c L_c |\psi\rangle = \sum_{j=1}^Q P_c L_c V_{c,j} (O - O_c) T_c |\psi_j\rangle.$$

The suffix leaves $D[c]$ unchanged, and P_c acts outside the oracle table. Commute L_c through the suffix and drop this projection after applying Cauchy–Schwarz. Using $\|u - v\|^2 \leq 2\|u\|^2 + 2\|v\|^2$ gives

$$p \leq 2Q \sum_{c,j} (\|P_c V_{c,j} O T_c |\psi_j\rangle\|^2 + \|P_c V_{c,j} O_c T_c |\psi_j\rangle\|^2) = 4Q^2 p_{\text{ext}}.$$

This is why we use the zeroed suffix. It commutes with L_c , so the bound concerns the extractor’s actual matching continuation. No assumption about an unchanged future transcript is needed.

The Fourier table is only used in the analysis. The extractor implements the zeroed suffix by comparing labels with the measured c .

At an active adversarial call to R in (2), measuring c fixes the full triple to $w(c)$. Measuring and retaining that triple causes no further disturbance. Internal calls can be discarded when their persistent records rule out P_c in both continuations. In particular, an honest key reveal at c records a key reveal at the tested sid before calling R , which excludes a permitted Test at c later.

Replacing the Test seed. We separate two correlations: the Test seed may depend on queries made before Test, and later queries may reveal the seed used at Test. The first hop removes the former dependence by replacing the oracle cell; the second removes the latter by restoring the original future oracle. Only then is the independent-seed PRF hop available.

Fix one of the guesses and embeddings specified below. Stop with a fair guess before answering a reveal of an embedded unknown secret. At Test, also reject an incorrect target or source guess, a missing required publication, or any prior key-reveal flag on a session with the target sid . These checks use the current classical transcript. They do not predict the final exposure case.

Run the preceding extractor up to Test, before its key evaluation. Let P_c require these prefix checks and that the tested sid is c . It also checks the public-key embedding and records the randomizers already delivered to the selected sessions. A successful active adversarial call yields both $w(c)$ and, through the modified continuation, these randomizers. The decoders below therefore give a Q-GA-GDH solver $\mathcal{B}_{\text{pre}}$, and

$$p \leq 4Q^2 \text{Adv}_{\text{Q-GA-GDH}}(\mathcal{B}_{\text{pre}}).$$

In the first hybrid, at a permitted Test swap the cell $R(c)$ with a fresh independent cell; retain the old cell in an unused environment register. Both Test and subsequent oracle calls now use the new cell. This swap is the identity when $D[c] = 0$, and it is not performed on rejected branches. Its change to the state has norm at most $2\sqrt{p}$, hence changes any output probability by at most $2\sqrt{p}$. Equivalently, this hybrid samples an independent seed t at Test and subsequently answers calls at c with t .

In the second hybrid, Test still uses t , but subsequent calls use the original R , including at c . The two post-Test executions start with the same state; their fixed oracle functions are $R[c \mapsto t]$ and R . The initial quantum state may be correlated with R, c, t . We use the fixed-function hybrid proof underlying ordinary O2H [4, Theorem 3], with this same state in both runs. For each fixed choice of the functions and classical transcript, purify that state. If $|\phi_j\rangle$ is the state before post-Test call j using the original R , the oracle difference is supported on the projection T_c onto an active call at c . Telescoping and averaging therefore bound the change in any output probability by

$$2 \sum_{j=1}^Q \mathbb{E} \|T_c |\phi_j\rangle\| \leq 2Q \sqrt{\frac{1}{Q} \sum_{j=1}^Q \mathbb{E} \|T_c |\phi_j\rangle\|^2} = 2Q \sqrt{p_{\text{post}}}.$$

This gives the bound without treating the correlated quantum state as the classical auxiliary input in the cited theorem. At a measured active adversarial call at c , the transcript already contains the required randomizers and the valid triple gives an immediate Q-GA-GDH solution. Before an honest key reveal at c , the game stops with a fair guess: the reveal would irreversibly invalidate freshness. The same rule applies if a session with an earlier reveal flag later acquires $\text{sid} = c$. Consequently the two seed hops cost at most

$$4Q \sqrt{\text{Adv}_{\text{Q-GA-GDH}}(\mathcal{B}_{\text{pre}})} + 2Q \sqrt{\text{Adv}_{\text{Q-GA-GDH}}(\mathcal{B}_{\text{post}})}. \quad (4)$$

Finally replace $\text{PRF}_t(c)$ in Test by a uniform session key. The PRF reduction samples all group-action secrets itself and implements the ordinary KDF directly, using a sufficiently independent function.

It computes all other session keys honestly and uses its classical PRF challenge oracle only for the Test value. This is exactly the second hybrid's observable distribution, by (2). In particular, this PRF reduction needs no decision oracle. The hidden PRF key is not programmed into the KDF, and no reveal requires it. Accidental equality with another seed needs no extra collision term: the PRF reduction evaluates that other seed normally in both of its challenge worlds.

Every hybrid returns the adversary's guess only on the selected, finally fresh case and uses a fair guess otherwise. After the PRF hop, the entire visible execution is independent of the Test bit, including the event selecting that case. Its signed advantage is therefore zero. If ϵ_g bounds the success probabilities of the gap reductions above and ϵ_p bounds the PRF advantage, the absolute signed advantage for a guessed experiment is at most

$$\Gamma := 6Q\sqrt{\epsilon_g} + \epsilon_p. \quad (5)$$

Neither ephemeral state is exposed

Lemma 1. *There are QPT gap-CDH and PRF reductions for which*

$$|\delta_I| \leq N^2(6Q\sqrt{\epsilon_{g,I}} + \epsilon_{p,I}),$$

where $\epsilon_{g,I}$ is the larger success probability of the two Q-GA-GDH reductions and $\epsilon_{p,I}$ is the PRF advantage.

Proof. Guess an ordered pair of session indices (T_0, S_0) uniformly from the N^2 possibilities. A repeated index fails. The guess is correct when $T = T_0$ and $S(T) = S_0$. Given a computational challenge $(E, [\alpha] \star E, [\beta] \star E)$, use $[\alpha] \star E$ as the raw ephemeral public value of T_0 and $[\beta] \star E$ as that of S_0 . Their unknown raw secrets are thus $[\alpha]$ and $[\beta]$. Sample all other secrets, including every signing and long-term key, honestly.

For a selected session receiving randomizer $[\tau]$, compute its final public element by applying $[\tau]$ to the challenge public element, and sign that public element with the known signing key. Record the randomizer. These operations work in either role and in either order. The KDF simulation above answers all session-key queries, including exchanges of S_0 with a different peer. Other state and long-term reveals are answered normally. Stop before answering a state reveal of T_0 or S_0 , even if it would return $\perp$: their exposure flags must both remain zero in the selected case.

At Test check the guess and source publication. If the target and source randomizers are $[\tau_T], [\tau_S]$, respectively, then a valid target triple satisfies

$$w_3 = [\alpha\beta\tau_T\tau_S] \star E.$$

Both the pre-Test and post-Test extractors return $[\tau_T\tau_S]^{-1} \star w_3 = [\alpha\beta] \star E$. The pre-Test extractor obtains the randomizers from its checked continuation; it need not know them at the measured query. The equality holds even if the two final public elements coincide.

For clarity about the guessing loss, first define the guessed experiment with all secrets sampled honestly and with the same abort rules. Its final guess is fair unless the guess is correct and C_1 holds. On every such real execution exactly one of the N^2 guesses is correct, and neither protected state was revealed. Its signed advantage is therefore exactly δ_I/N^2 . Embedding two independent challenge secrets gives the same distribution up to the permitted stops. Apply (5) and multiply by N^2 .

One side's ephemeral state is exposed

Lemma 2. *There are QPT gap-CDH and PRF reductions for which*

$$\begin{aligned} |\delta_{II,t}| &\leq N^2(6Q\sqrt{\epsilon_{g,t}} + \epsilon_{p,t}), \\ |\delta_{II,s}| &\leq N\mu(6Q\sqrt{\epsilon_{g,s}} + \epsilon_{p,s}). \end{aligned}$$

For each row, ϵ_g is the larger success probability of its two Q-GA-GDH reductions and ϵ_p is its PRF advantage.

Proof. The two cases protect different pairs of secrets, so we specify their embeddings separately. Signing keys are generated independently and are known in both reductions.

In $C_{\text{II},t}$, guess (T_0, S_0) among N^2 ordered session pairs. Embed $[\alpha] \star E$ as the raw ephemeral public value of S_0 , and $[\beta] \star E$ as the long-term action public key of the owner of T_0 . Generate the target's ephemeral secret and every other secret normally. Stop before a state reveal of S_0 or a long-term reveal of the target owner. These stops cannot occur on a correctly guessed execution in $C_{\text{II},t}$: all source states are unexposed, and exposing the target state forces its owner's long-term key to remain hidden by freshness. In particular $S_0 \neq T_0$ on every selected execution.

At Test require $T = T_0$ and $S(T) = S_0$. Let $[\mathbf{r}_\mathfrak{E}]$ be the recorded source randomizer. If the target is an initiator, its mixed share $w_2 = [\alpha\beta\mathbf{r}_\mathfrak{E}] \star E$; if it is a responder, the same equality holds for w_1 . The decoder applies $[\mathbf{r}_\mathfrak{E}]^{-1}$. This embedding does not depend on which role the source used, which peer it named, or whether it has finished its own exchange.

In $C_{\text{II},s}$, guess a target session T_0 and an honest peer party P_{j_0} among $N\mu$ possibilities. Embed $[\alpha] \star E$ as the target's raw ephemeral public value and $[\beta] \star E$ as that peer's long-term action public key. Generate every source ephemeral secret normally. Stop before a state reveal of T_0 or a long-term reveal of P_{j_0} . The target state must be unexposed in this case. Since some source state is exposed, the source freshness clause keeps the peer's long-term key hidden. Thus neither stop removes a correctly guessed selected execution.

At Test require $T = T_0$, peer P_{j_0} , and a nonempty source set. Let $[\mathbf{r}_\mathfrak{T}]$ be the target randomizer. For an initiator, $w_1 = [\alpha\beta\mathbf{r}_\mathfrak{T}] \star E$; for a responder, this holds for w_2 . Apply $[\mathbf{r}_\mathfrak{T}]^{-1}$ to recover the challenge target. No particular source needs to be guessed in this case.

In both reductions, public values are rerandomized and signed as in the preceding proof, and the exact KDF simulation handles every unrelated session. All remaining reveals are answered with the known secrets. The final case check is performed at the end, since exposures can occur after Test; it is not used to avoid simulating earlier queries. The first guessed experiment has signed advantage $\delta_{\text{II},t}/N^2$ and the second has $\delta_{\text{II},s}/(N\mu)$, by the same coupling as in Case I. Apply (5) to each experiment.

Accepted shares without an honest source

Lemma 3. *For every QPT adversary $\mathcal{A}$, there is a QPT EU-CMA adversary $\mathcal{B}$ such that*

$$\Pr[C_{\text{IV}}] \leq \mu \cdot \text{Adv}_{\text{EU-CMA}}^\Sigma(\mathcal{B}), \quad |\delta_{\text{IV}}| \leq \frac{1}{2}\mu \cdot \text{Adv}_{\text{EU-CMA}}^\Sigma(\mathcal{B}).$$

Proof. For each honest party P_j , maintain the set M_j of messages signed by any of its sessions in either role. Insert a message when its honest signed publication is produced. Let F be the first acceptance at which the peer P_j is uncorrupted and the verified message m is absent from M_j . An execution in C_{IV} has such an acceptance: its peer was uncorrupted until target acceptance by freshness, and its source set is empty. Thus $C_{\text{IV}} \subseteq F$.

The reduction guesses j uniformly from $[\mu]$ and assigns its EU-CMA challenge verification key to P_j . It generates every group-action secret, every ephemeral secret, and the other signing keys honestly. Whenever a session of P_j signs, in either role, the reduction submits that exact message to its classical signing oracle and records it in M_j . All remaining protocol operations and reveal queries are simulated honestly, except that the reduction stops with failure if $\text{RevLTK}(j)$ is issued before a forgery is obtained.

At the first event F , if its peer is P_j , the reduction outputs the accepted pair (m, σ) . Verification succeeds and $m \notin M_j$, so the pair is an EU-CMA forgery. This output occurs before the peer's corruption; no later secret-key reveal needs to be answered. Conditioned on guessing the peer of the first event F , the simulation has not stopped prematurely and has the real distribution up to that event. Consequently $\Pr[F] \leq \mu \cdot \text{Adv}_{\text{EU-CMA}}^\Sigma(\mathcal{B})$.

For completeness, the reduction can implement the independent KDF oracle by a fixed $2Q$ -wise independent function, where Q bounds all KDF queries by the adversary and the honest simulation together. Use an injective encoding of strings up to the polynomial input-length bound. Its behavior for this bounded execution is identical to that of a random function [44]. The reduction knows all DH secrets, so it computes honest KDF inputs directly; no reprogramming or decision oracle is needed here.

Finally, $|\Pr[W \wedge C] - \frac{1}{2}\Pr[C]| \leq \frac{1}{2}\Pr[C]$ for any event C , proving the contribution bound. The argument covers both roles and requires a new signed message, not a new signature on an old message. It does not identify honest replays with forgeries.

Combining the cases. Let ϵ_g be the maximum success probability of the six Q-GA-GDH reductions, ϵ_p the maximum advantage of the three PRF reductions, and ϵ_{sig} the signature reduction's advantage. Combining the lemmas gives

$$\text{Adv}_{G_1}^A \leq (2N^2 + N\mu)(6Q\sqrt{\epsilon_g} + \epsilon_p) + \frac{1}{2}\mu\epsilon_{\text{sig}}.$$

This proves the bound in Theorem 2. Under the stated assumptions each reduction's advantage is negligible. Since N, Q are polynomially bounded and a square root of a negligible function is negligible, the AKE advantage is negligible as well.

A.3 Proof of Corollary 1

The proof of corollary 1 uses the same KDF simulation and extraction argument as Theorem 2. The additional work is to simulate the permitted subversions and disclosures. In particular, we cannot identify a client's raw secret with the secret represented by the firewall's outgoing element: a state query can distinguish them.

Write $\mathcal{B}$ for the statically selected set of subverted clients, and $N = \mu\ell$. Count in q_s the external adversary's KDF queries and all KDF queries made by subverted code, and put $Q = q_s + N + 1$. The running times of the reductions include execution of that code. The adversary and replacement clients are classical PPT algorithms; we use the QROM reductions of Theorem 2 for these algorithms without asserting security against quantum subversions.

All target, source and peer guesses below are made before setup. Any challenge action public element or signature verification key is placed in the registry before the adversary selects its replacements. The resulting public keys have the honest distribution. A guess that is incompatible with the subsequently selected subversion set stops.

Replacing clients whose state is not disclosed. For a fixed set $U \subseteq \mathcal{B}$, consider an experiment that stops immediately before the first `RevState` query to any session of any client in U . A stop replaces the final guess by an independent fair bit. Replace every client in U , together with its firewall, by the following simulator. For each new ephemeral publication, sample a uniform class $[z]$ and send $Z = [z] \star E$ to the server. On receiving the server's actual randomizer $[\tau]$, publish $X = [\tau] \star Z$ with its prescribed valid signature. Process all other messages, acceptance decisions, session identifiers, and key outputs according to the prescribed protocol, using this final public element. The simulator keeps the original registered long-term keys.

Lemma 4. *This replacement preserves the joint distribution of the adversary's state, all protocol messages and query replies, and the public publication and exposure records in the stopped experiment.*

Proof. Fix the registered keys and all random functions. Consider the next raw public element X_0 supplied by a protected client. Explainability ensures that X_0 is a valid action element. Its choice may depend on the client's entire private history, including retained keys, earlier firewall replies, and private oracle queries. The fresh firewall mask is sampled after X_0 is fixed and independently of that history. By regularity, $[\alpha] \star X_0$ is uniform on $\mathcal{E}$. Its conditional distribution is the same for every X_0 , so it is independent of the preceding history and equals the distribution of the simulator's Z .

Couple these outgoing elements. The adversary receives the same messages and has the same state, so its adaptive server reply $[\tau]$ has the same joint distribution in both experiments. The final element is $[\tau] \star Z$. To determine the client's responses, apply explainability to its entire invocation sequence, including the inputs delivered by the firewall. Every explaining tape produces X_0 as the raw public element. Since the action is free, all these tapes use the same raw action class, and the received randomizer fixes the same effective class. Honest signing produces a valid signature on the resulting element, whose value is fixed by uniqueness. Validation, the remaining messages and aborts follow the honest protocol for the same inputs, including invalid messages.

The same tape explains the local session results. Hence acceptance, the peer, role and `sid` agree with the protocol, and acceptance with `sid = c` gives the key $\text{PRF}_{H(c, w(c))}(c)$, where $w(c)$ is the unique valid DH triple of eq. (2). All explaining tapes give this value under the fixed random functions. We use only these consequences of explainability; we neither recover a tape nor condition the firewall mask on one.

A long-term query returns the same fixed registered keys. A state query to a protected client stops both experiments before returning its memory, including when the query would have returned an empty value. Other clients run their actual code with the same inputs, private states, and coins. Their state and session-key replies therefore agree as well. The protected clients' unused private oracle reads do not modify the fixed random functions, and these reads and their running times are not additional communication interfaces. Induction over the successive interactions proves the claim for the fixed functions and keys. Averaging proves the stated distributional equality.

The stop prevents the replaced clients' private memories from being returned. The same argument applies to an honest selected session up to its first state query: its masked initial element can be generated directly as $[3] \star E$. We use this observation only for selected honest sessions whose state must remain unexposed. Every other honest session runs the actual wrapped protocol, with its actual raw secret and firewall mask. Its state replies use those representations, rather than a different representative of their product.

The KDF simulation and the case partition. Use the exact context oracle of eq. (2). An honest or replaced client stores its prescribed key symbolically, evaluating the seed $R(c)$ only when a key must be returned. This postponement does not affect later messages, and no disclosed state of such a client contains the postponed value. For an honest client this follows from its specified state interface and erasure; for a replaced client it follows from the stop above.

All other subverted clients are run literally. Their raw choices may depend on earlier keys, so their private KDF computations cannot be postponed. These computations use the ordinary tuple interface $F(c, w)$ and are included in q_s . In each reduction below, the replaced set contains every subverted client among the guessed target owner and peer. A literally executed client's prescribed sid contains its own identity, which lies outside these two identities; its prescribed key computation therefore cannot have the tested sid . An additional private oracle query may have that sid , but it supplies a full tuple through F and is an ordinary extractable query. No literal subversion receives privileged access to R .

All publications are recorded after the firewall, including those of subverted clients. The source set retains every matching publication before target acceptance, irrespective of the source's role, peer, sid , or completion. Explainability ensures that every accepted session with the same sid has the same key. Thus the agreement step, the earliest-source rule $S(T)$, and the signed contributions δ_C in the proof of Theorem 2 apply unchanged. Use its disjoint cases $C_I, C_{II,t}, C_{II,s}, C_{IV}$, including the additional final freshness clause for subverted clients. The case with both target and source state exposed is still unfresh.

Neither ephemeral state exposed. Guess an ordered target and earliest source (T_0, S_0) among N^2 session pairs. Guesses whose owners coincide have zero selected contribution and stop. Let U contain the subverted clients among these two owners and apply Lemma 4. Also stop before any state query to T_0 or S_0 when its owner is honest, including queries before publication or after erasure. No such stop occurs on a correctly guessed fresh execution in C_I .

Given a challenge $(E, [\gamma] \star E, [\delta] \star E)$, use its two public elements as the selected sessions' masked initial elements. Their distributions are independent and uniform, even if the two replaced clients choose correlated raw secrets. Generate all long-term keys normally. Record the actual server randomizers $[\mathbf{r}_T], [\mathbf{r}_S]$ and obtain final elements by their public action. The selected honest sessions use the preceding simulation up to their state stops; no unknown private mask is supplied to replacement code. For a valid tested triple,

$$w_3 = [\gamma \delta \mathbf{r}_T \mathbf{r}_S] \star E.$$

Both extractors from the base proof recover the challenge solution by applying $[\mathbf{r}_T \mathbf{r}_S]^{-1}$ to w_3 . The pre-Test extractor obtains these randomizers from its checked continuation, not from the measured query alone. The identity also holds when the final public elements coincide.

Target state exposed, source states unexposed. For $C_{II,t}$, guess (T_0, S_0) among N^2 session pairs. The target owner must be honest: any state query to a subverted target owner would violate the additional freshness clause. Reject guesses that do not have this property. Replace the source owner if subverted, and stop before its first state query to any session. If honest, stop before a state query to S_0 . Also stop before a long-term query to the target owner.

Embed $[\gamma] \star E$ as S_0 's masked initial element and $[\delta] \star E$ as the target owner's long-term action public key. All signing keys and all other action secrets are known. In particular, the honest target and its owner's other sessions have actual sampled raw secrets and firewall masks, so their permitted state replies are available. Their unknown long-term action secret is needed only for DH computations, which the exact context oracle handles. For an initiating target the decoder uses w_2 ; for a responding target it uses w_1 . In either case the selected component equals $[\gamma\delta\mathbf{r}_\mathbf{E}] \star E$, so applying $[\mathbf{r}_\mathbf{E}]^{-1}$ solves the challenge. The source's own role and intended peer do not matter. Freshness justifies all stops: every source state is unexposed, and exposure of the target state requires its long-term key to remain unexposed.

Source state exposed, target state unexposed. For $C_{\text{II},s}$, guess T_0 and its peer P_{j_0} among $N\mu$ choices. The peer must be honest, since a state query to any of its source sessions would otherwise violate the additional clause. Reject guesses with a subverted peer or coincident owners. Replace the target owner if subverted, stopping before every state query to that client. If honest, stop before any state query to T_0 . Also stop before a long-term query to P_{j_0} .

Embed $[\gamma] \star E$ as T_0 's masked initial element and $[\delta] \star E$ as P_{j_0} 's long-term action public key. The peer's honest sessions use known raw secrets and masks; their actual state replies can therefore be returned. The exact context oracle supplies their prescribed keys. For an initiating target the decoder uses w_1 ; for a responding target it uses w_2 . The selected component is $[\gamma\delta\mathbf{r}_\mathbf{T}] \star E$, and the decoder removes $[\mathbf{r}_\mathbf{T}]$. No particular source is guessed. The nonempty source set, the target state flag, and the peer's long-term flag are checked as in the base proof. Since some source state is exposed, freshness keeps that peer's long-term key hidden. In both mixed cases an unknown long-term action secret thus belongs to an honest client; the reduction never passes it to arbitrary replacement code.

Applying the shared extraction argument. For each DH case, first define the guessed experiment with all secrets known, the stated stops, and a fair final guess unless the guessed target, peer or earliest source, and final case are correct. Its signed contribution is exactly δ_C/M_C , where $M_I = M_{\text{II},t} = N^2$ and $M_{\text{II},s} = N\mu$. There is exactly one successful guess for each selected fresh execution, and none of the added stops removes it. Only then replace the protected clients and embed the independent challenge elements. The preceding distributional equalities justify these steps without assuming independent raw secrets from subverted code.

The pre-Test extractor uses the same prefix checks as the base proof, including the selected publications and randomizers. Internal seed reads returning a session key record their key-reveal flags first, so they cannot qualify at the tested sid. Literally executed subversions' queries use the tuple interface and yield $w(c)$ on an active measured branch. They are therefore handled by the same extractor, even if their next computation depends on that answer. The modified continuation runs that code on its modified answers; eq. (3) does not require the original continuation to survive measurement.

After Test, stop before any reply that would invalidate the selected experiment, including a state query to a protected subverted client or a key query at the tested sid. The common-state O2H calculation then applies with the same decoders. For the PRF hop, sample all action secrets and firewall masks normally and implement an ordinary independent KDF. Run the unrelated replacements literally and the protected clients by their prescribed public computations and outputs. Only Test uses the independent PRF challenge. This reduction needs no decision oracle. The query bound includes the literal computations and at most N postponed session seed evaluations. Consequently eq. (5) gives

$$|\delta_C| \leq M_C(6Q\sqrt{\epsilon_{g,C}} + \epsilon_{p,C}) \quad (C \in \{\text{I}, \text{II}, t, \text{II}, s\}).$$

An accepted share without a source. For each registered client P_j , let M_j contain every message in its recorded signed publications, in either role. Let F be the first source-free acceptance whose peer P_j has not been queried by RevLTK and, if $P_j \in \mathcal{B}$, has not had a state query to any session. A fresh execution in C_{IV} contains such an event. The extra qualification is necessary: an earlier acceptance involving an already state-exposed subverted peer need not yield a signature forgery.

Guess the peer of F uniformly among the μ clients and give it the EU-CMA challenge verification key. Generate every action secret and all other signing keys normally. If the guessed peer is subverted, replace it as in Lemma 4, stopping before its first state query. If honest, run its honest code. Use the signing oracle for each prescribed publication, inserting its message in M_j . Stop before the guessed peer's long-term query if no forgery has yet been obtained. All other clients run their actual code with known

keys. Protected code's unobserved private signing computations have been removed by the replacement, so they introduce no additional signing-oracle messages.

Conditioned on guessing the peer of the first event F , the simulation has the real distribution up to that event and has not stopped. Its accepted pair (m, σ) verifies and satisfies $m \notin M_j$, so it is an EU-CMA forgery. Extraction precedes any later permitted long-term disclosure. The reduction knows all action secrets and implements the independent KDF directly, including all queries of the literal replacements; no decision oracle is required. Therefore

$$|\delta_{IV}| \leq \frac{1}{2} \Pr[C_{IV}] \leq \frac{\mu}{2} \epsilon_{\text{sig}}.$$

Summing the four signed contributions proves

$$\text{Adv}_{H_W}^{\text{AKE-FS}}(\mathcal{A}) \leq (2N^2 + N\mu)(6Q\sqrt{\epsilon_g} + \epsilon_p) + \frac{\mu}{2} \epsilon_{\text{sig}},$$

where ϵ_g bounds the six gap reductions and ϵ_p the three PRF reductions above. The replacement introduces no statistical loss. The reduction uses the same computational assumptions as the base theorem; its additional premises are the explicit subversion and disclosure conditions of the firewalled experiment.

A.4 Proof of Theorem 4

Theorem 4. *The binary relation $<_\phi$ on $\text{Valid}(\phi)$ is a total strict order.*

Proof. Let $\phi_1, \phi_2, \phi_3 \in \text{Valid}(\phi)$.

Irreflexivity. If $\phi_1 <_\phi \phi_1$, then since $x_{\phi_1(P_E)} = x_{\phi_1(P_E)}$, we must have $y_{\phi_1(P_E)} < y_{\phi_1(P_E)}$ which is impossible. Hence $\phi_1 \not<_\phi \phi_1$.

Transitivity. Assume that $\phi_1 <_\phi \phi_2$ and $\phi_2 <_\phi \phi_3$. If $x_{\phi_1(P_E)} <_2 x_{\phi_3(P_E)}$, then $\phi_1 <_\phi \phi_3$. If not, then since $\phi_i <_\phi \phi_j$ implies that $x_{\phi_i(P_E)} = x_{\phi_j(P_E)}$ or $x_{\phi_i(P_E)} <_2 x_{\phi_j(P_E)}$, we must have $x_{\phi_1(P_E)} = x_{\phi_3(P_E)}$. From $x_{\phi_1(P_E)} = x_{\phi_3(P_E)}$, $\phi_1 <_\phi \phi_2$ and $\phi_2 <_\phi \phi_3$, we get that $y_{\phi_1(P_E)} <_2 y_{\phi_2(P_E)}$ and $y_{\phi_2(P_E)} <_2 y_{\phi_3(P_E)}$, which implies that $y_{\phi_1(P_E)} <_2 y_{\phi_3(P_E)}$. Hence $\phi_1 <_\phi \phi_3$.

Comparability. Assume that $\phi_1 \not<_\phi \phi_2$ and $\phi_2 \not<_\phi \phi_1$. Then we must have that $x_{\phi_1(P_E)} = x_{\phi_2(P_E)}$, as $x_{\phi_1(P_E)} \neq x_{\phi_2(P_E)}$ would imply that either $x_{\phi_1(P_E)} <_2 x_{\phi_2(P_E)}$ and $\phi_1 <_\phi \phi_2$, or $x_{\phi_2(P_E)} <_2 x_{\phi_1(P_E)}$ and $\phi_2 <_\phi \phi_1$. From $x_{\phi_1(P_E)} = x_{\phi_2(P_E)}$, $\phi_1 \not<_\phi \phi_2$ and $\phi_2 \not<_\phi \phi_1$, we must have $y_{\phi_1(P_E)} = y_{\phi_2(P_E)}$, as $y_{\phi_1(P_E)} \neq y_{\phi_2(P_E)}$ would imply that either $y_{\phi_1(P_E)} <_2 y_{\phi_2(P_E)}$ and $\phi_1 <_\phi \phi_2$, or $y_{\phi_2(P_E)} <_2 y_{\phi_1(P_E)}$ and $\phi_2 <_\phi \phi_1$. Hence $x_{\phi_1(P_E)} = x_{\phi_2(P_E)}$ and $y_{\phi_1(P_E)} = y_{\phi_2(P_E)}$, implying that $\phi_1(P_E) = \phi_2(P_E)$. Since $\phi_1, \phi_2 \in \text{Valid}(\phi)$, then there exists $\eta \in \text{Aut}(F_x)$ such that $\phi_2 = \eta \circ \phi_1$, which implies that $T = \eta(T)$ where $T = \phi_1(P_E)$. Since $\deg(\phi_1) = q = \deg(\phi_2)$ and $\gcd(N, q) = 1$, then $T = \phi_1(P_E)$ has order N . Hence, $(\eta - \text{Id})(T) = 0$ and $\deg(\eta - \text{Id})$ is divisible by N . Let us assume that $\eta \neq \text{Id}$. Following [42, Lemma 3.1], we have that

$$|\deg(\eta - \text{Id}) - \deg(\eta) - 1| \leq 2\sqrt{\deg(\eta)}.$$

Since η is an automorphism, then $\deg(\eta) = 1$ and $|\deg(\eta - \text{Id}) - 2| \leq 2$, meaning that $\deg(\eta - \text{Id}) \leq 4$. This contradicts the fact that N divides $\deg(\eta - \text{Id})$, as $N > q$ and q is an exponentially large prime. We must then have $\eta = \text{Id}$ and $\phi_1 = \phi_2$. $\square$

B Security Model for Signal-Conforming AKE Protocols

We adapt the Signal-Conforming authenticated key-exchange (AKE) model of [29] to iX3DH. Besides full-session partnering, we track the honest sessions that supplied the ephemeral shares used by a tested session. This adds exposure restrictions for pending sessions and replayed shares, as specified below. The secrecy experiment tests exchanges between distinct parties. Other sessions, including sessions a party starts with itself, remain available to the adversary.

Execution Environment. We consider a system of μ parties $P_1, \dots, P_\mu$. Each party P_i is represented by a set of ℓ oracles $\{\pi_i^1, \dots, \pi_i^\ell\}$, where each oracle corresponds to a single execution of a protocol, and $\ell \in \mathbb{N}$ is the maximum number of protocol sessions per party. Each oracle is equipped with fixed randomness but is otherwise deterministic. Each oracle π_i^s has access to the long-term key pair $(\text{pk}_i, \text{sk}_i)$ of P_i and the public keys of all other parties, and maintains a list of the following local variables:

- rand_i^s is the randomness hard-wired to π_i^s ;
- sid_i^s stores the identity of the session as specified by the protocol;
- Pid_i^s stores the identity of the intended communication partner;
- $\Psi_i^s \in \{\perp, \text{accept}, \text{reject}\}$ indicates whether oracle π_i^s has successfully completed the protocol execution and accepted the resulting key;
- k_i^s stores the session key computed by π_i^s ;
- state_i^s holds the (secret) session-state values and intermediary results required by the session;
- $\text{role}_i^s \in \{\perp, \text{init}, \text{resp}\}$ indicates π_i^s 's role during the protocol execution.

For each oracle π_i^s , these variables, except the randomness, are initialized to $\perp$. An AKE protocol is executed interactively between two oracles. An oracle that first sends a message is called an initiator ($\text{role} = \text{init}$) and a party that first receives a message is called a responder ($\text{role} = \text{resp}$). The computed session key is assigned to the variable k_i^s if and only if π_i^s reaches the accept state, that is, $\text{k}_i^s \neq \perp \iff \Psi_i^s = \text{accept}$.

Partnering. To exclude trivial attacks in the security model, we need to define a notion of “partnering” of two oracles. Intuitively, this dictates which oracles can be corrupted without trivializing the security game.

Definition 16 (Partnering). *For any $(i, j, s, t) \in [\mu]^2 \times [\ell]^2$ with $i \neq j$, we say that oracles π_i^s and π_j^t are partners if (1) $\text{Pid}_i^s = j$ and $\text{Pid}_j^t = i$; (2) one oracle has role **init** and the other has role **resp**; and (3) $\text{sid}_i^s = \text{sid}_j^t \neq \perp$.*

An undefined session identifier does not establish a partnership. In particular, two pending sessions do not become partners merely because their identifiers both equal $\perp$.

Party identities and session identifiers play different roles here. Partners belong to distinct parties but have the same sid . Two sessions owned by one party can also have the same sid ; they are not partners, but the key-reveal restriction below covers them. We do not assume that a session has a unique partner under adversarial scheduling. The condition $i \neq j$ in the partnering definition does not itself exclude testing a self session: such a session would simply have no partner. We therefore exclude self sessions explicitly in **Test**, while allowing them to run and to supply shares to other sessions.

Sources of ephemeral shares. The challenger records every signed final ephemeral element published by an honest session, together with the session index and the order of publication. These records are kept after the session completes or erases its state. For an accepted session $T = \pi_i^s$ with peer P_j , let $\text{Src}(T)$ contain every session of P_j that published the peer ephemeral element accepted by T before T accepted. Equality here means equality of the canonically encoded signed message, not of the signature bytes.

A source need not have completed, have the same sid , or name P_i as its peer. Both signing roles are included, since the signatures do not distinguish roles. All matching sources are included. The set is fixed at target acceptance; subsequent publications cannot establish an earlier honest origin. This relation records the source of a public share, not agreement on a session key.

Session state and erasure. Before sampling an ephemeral secret, a session's state is $\perp$. While awaiting the server's randomizer, its state contains the raw ephemeral secret. After applying the randomizer, the session retains the resulting secret until it completes. The server exchange consists of separate message deliveries: state reveals are allowed between them. In particular, revealing the raw secret also exposes the final secret once the public randomizer is known.

Signature verification, the DH and KDF computations, key derivation, and erasure take place within the final local processing step, without intervening adversarial queries. The session then retains its public transcript and session key, and erases its ephemeral secret, DH shares, and intermediate KDF key. A rejected session also erases its temporary secrets. Later state reveals return $\perp$; session keys are accessed through **RevSessKey**. Alice's pending session retains its secret while she is offline. Her sid is assigned only when she receives Bob's response and completes; Bob assigns his sid and accepts in the step producing that response.

An exposure flag records every **RevState** query and is never reset. Thus erasure does not undo an earlier exposure. We conservatively keep the flag even for a query returning $\perp$. Long-term keys are

accessed separately through RevLTK and are not included in session state. For correctness, we require that two oracles executing the AKE without adversarial interaction derive identical session-identifiers. We also require that two such oracles reach the accept state and derive identical session keys except with negligible probability. We call a set $S \subseteq ([\mu] \times [\ell])^2$ to have a valid pairing if the following properties hold:

- For all $((i, s), (j, t)) \in S, i \leq j$.
- For all $(i, s) \in [\mu] \times [\ell]$, there exists a unique $(j, t) \in [\mu] \times [\ell]$ such that $i \neq j$ and either $((i, s), (j, t)) \in S$ or $((j, t), (i, s)) \in S$.

In other words, a set with a valid pairing S partners off each oracle π_i^s and π_j^t in a way that the pairing is unique and no oracle is left out without a pair. We define correctness of an AKE protocol as follows, mainly from [22].

Definition 17 (AKE Correctness). *An AKE protocol Π_{AKE} is $(1 - \delta)$ -correct if for any set with a valid pairing $S \subseteq ([\mu] \times [\ell])^2$, when we execute the AKE protocol without adversarial interaction between all the oracle pairs included in S , it holds that*

$$(1 - \delta) \leq \Pr \left[\pi_i^s \text{ and } \pi_j^t \text{ are partners} \wedge \Psi_i^s = \Psi_j^t = \text{accept} \right. \\ \left. \wedge k_i^s = k_j^t \neq \perp \text{ for all } ((i, s), (j, t)) \in S \right],$$

where the probability is taken over the randomness used in the oracles.

Security Game. We define the security of an AKE protocol Π_{AKE} via a game $G_{\Pi_{\text{AKE}}}^{\text{FS}}(\mu, \ell)$ played between an adversary $\mathcal{A}$ and a challenger C . More formally, the security game is parameterized by two integers μ (the number of honest parties) and ℓ (the maximum number of protocol executions per party), and proceeds as follows:

Setup: C first chooses a challenge bit $b \in \{0, 1\}$ at random. C then generates the public parameter of Π_{AKE} and μ long-term key pair $\{(\text{pk}_i, \text{sk}_i) \mid i \in [\mu]\}$, and initializes the collection of oracles $\{\pi_i^s \mid i \in [\mu], s \in [\ell]\}$. C runs $\mathcal{A}$ providing the public parameter and all the long-term public keys $\{\text{pk}_i \mid i \in [\mu]\}$ as input.

All protocol, registration, reveal, and Test queries are classical. The adversary has quantum access to the KDF throughout both phases. A party identity has one immutable registered public key. Queries with invalid oracle indices or malformed query syntax return $\perp$ and do not change the experiment. Message contents supplied to Send are instead validated and processed by the protocol, including its rejection and erasure rules. Each session slot can be started only once, and a defined sid is retained thereafter. Reveal records are permanent; a well-formed reveal of an existing session slot is recorded even when its answer is $\perp$.

Phase 1: $\mathcal{A}$ adaptively issues the following queries any number of times in an arbitrary order:

- **Send**(i, s, m): This query allows $\mathcal{A}$ to send an arbitrary message m to oracle π_i^s . The oracle will respond according to the protocol specification and its current internal state. To start a new oracle, the message m takes a special form:
 $\langle \text{START: role}, j \rangle$ is accepted only for an unused slot, a role in $\{\text{init}, \text{resp}\}$, and a peer already in the public-key registry; otherwise it returns $\perp$. The choice $j = i$ is allowed. C initializes π_i^s in the role role , having party P_j as its peer, that is, C sets $\text{Pid}_i^s := j$ and $\text{role}_i^s := \text{role}$. If π_i^s is an initiator (i.e., $\text{role} = \text{init}$), then C returns the first message of the protocol. Later messages are processed only in the states permitted by the protocol; completed sessions cannot restart.
- **RevLTK**(i): For $i \in [\mu]$, this query allows $\mathcal{A}$ to learn the long-term secret key sk_i of party P_i . After this query, P_i is considered to be corrupted.
- **RegLTK**(i, pk_i): For a previously unused $i \in \mathbb{N} \setminus [\mu]$, this query allows $\mathcal{A}$ to register a new party P_i with public key pk_i satisfying the protocol's public-key validation rules. We do not require that the adversary knows the corresponding secret key. After the query, the pair (i, pk_i) is distributed to all other oracles. These parties are corrupted from registration and have no challenger-run session oracles. An attempted registration under an existing identity returns $\perp$.
- **RevState**(i, s): This query allows $\mathcal{A}$ to learn the session state state_i^s of oracle π_i^s . After this query, state state_i^s is said to be revealed.
- **RevSessKey**(i, s): This query allows $\mathcal{A}$ to learn the session key k_i^s of oracle π_i^s .

Test: Once $\mathcal{A}$ decides that Phase 1 is over, it issues the following special Test-query, which returns a real or random key depending on the challenge bit b .

- **Test**(i, s): If $(i, s) \notin [\mu] \times [\ell]$ or $\Psi_i^s \neq \text{accept}$ or $\text{Pid}_i^s = i$, C returns $\perp$. Else, C returns k_b , where $k_0 := k_i^s$ and $k_1 \leftarrow \mathcal{K}$ (where $\mathcal{K}$ is the session key space). Only when a key is returned is π_i^s said to be tested.

Phase 2: $\mathcal{A}$ adaptively issues queries as in Phase 1.

Guess: Finally, $\mathcal{A}$ outputs a guess $b' \in \{0, 1\}$. At this point, the validly tested oracle must be fresh. Such an oracle π_i^s with $\text{Pid}_i^s = j$ is fresh if all the following conditions hold:

- $\text{RevSessKey}(i, s)$ has not been issued;
- for every oracle π_h^t with $\text{sid}_h^t = \text{sid}_i^s \neq \perp$, $\text{RevSessKey}(h, t)$ has not been issued, including oracles owned by P_i ;
- P_i is not corrupted or state state_i^s is not revealed;
- for every partner π_j^t of π_i^s , P_j is not corrupted or state state_j^t is not revealed;
- for every partner π_j^s of π_i^s , at least one of state_i^s and state_j^s has not been revealed;
- for every $R \in \text{Src}(\pi_i^s)$, P_j is not corrupted or the state of R has not been revealed;
- for every $R \in \text{Src}(\pi_i^s)$, at least one of the states of π_i^s and R has not been revealed;
- if π_i^s has no partner oracle or $\text{Src}(\pi_i^s) = \emptyset$, then any corruption of P_j occurs only after π_i^s accepts.

If no oracle was validly tested, or the tested oracle is not fresh, C aborts the game and outputs a random bit b' on behalf of $\mathcal{A}$. We say that $\mathcal{A}$ wins the game if $b = b'$. The advantage of $\mathcal{A}$ in the security game above is defined as

$$\text{Adv}_{\Pi_{\text{AKE}}}^{\text{AKE-FS}}(\mathcal{A}) := \left| \Pr[b = b'] - \frac{1}{2} \right|.$$

Definition 18 (AKE Security). An AKE protocol Π_{AKE} is secure with perfect forward secrecy in this model if the advantage $\text{Adv}_{\Pi_{\text{AKE}}}^{\text{AKE-FS}}(\mathcal{A})$ is negligible for any QPT adversary $\mathcal{A}$.

Remark 3. These restrictions strengthen the freshness requirements of [29]; they therefore give a weaker security guarantee. Write $[\mathbf{u}]$, $[\mathbf{v}]$ for the effective ephemeral secrets and $[\mathbf{a}]$, $[\mathbf{b}]$ for the corresponding long-term secrets. Each of the pairs $([\mathbf{a}], [\mathbf{u}])$, $([\mathbf{b}], [\mathbf{v}])$, and $([\mathbf{u}], [\mathbf{v}])$ suffices to compute all three DH shares from public elements. The source conditions exclude these combinations even when the session supplying a share is still offline or has a different sid.

For example, Bob may accept while Alice’s sid is still undefined. Her publishing session is nevertheless in $\text{Src}(T)$, so revealing its state and later corrupting Alice makes Bob’s session unfresh. Long-term corruption after acceptance remains permitted when the relevant session states have not been revealed. No additional message or online confirmation is required. If there was no source at acceptance, a later matching session does not remove the requirement that the peer was uncorrupted until acceptance.

The Test restriction concerns only the identity of the tested peer: the secrecy claim does not cover a party exchanging a key with itself. Such sessions are still simulated and can be sources for another party’s test. Since every source of a valid test belongs to its distinct peer, its ephemeral secret occupies a different session from the target’s. This is the independence needed in the ephemeral–ephemeral reduction.

A fresh tested peer must be one of the μ initially honest parties. A party registered by the adversary has no honest source sessions and is already corrupted before it can be named as a peer, which violates the last freshness clause.

The session-key restriction covers all oracles with the tested sid, including two sessions owned by the same party. Such sessions compute the same key but are not partners under the definition above. This matters when the server uses exposed raw secrets to force equal final ephemeral shares. Revealing another copy of the tested key must make the test unfresh.

AKE with reverse firewalls. The preceding experiment describes honest client implementations. We now define its extension for subverted clients behind honest reverse firewalls. We use the explainability notion of Definition 1 and specify the disclosure rules below.

Definition 19 (Firewalled AKE experiment). Let Π_W denote iX3DH with an honest firewall at every challenger-run client. The experiment $\text{Game}_{\Pi_{\text{AKE}}}^W(\mu, \ell)$ uses the setup, queries, Test rule and winning condition of $G_{\Pi_{\text{AKE}}}^{\text{FS}}(\mu, \ell)$, with the following changes.

Subversions and communication. After honest key generation and publication of the immutable public keys, but before any session or exposure query, the PPT adversary selects a set $U \subseteq [\mu]$ and one classical stateful polynomial-time replacement for each client in U . These choices remain fixed. Each replacement receives its client's registered long-term keys. It may bias its randomness, use its private inputs, retain secrets, and share its own memory across that client's sessions. The adversary may know the replacement's initial auxiliary information; selecting it does not reveal the honestly generated long-term keys. Clients have separate private memories, and no replacement shares live memory with the external adversary. Communication takes place only through the protocol and experiment interfaces. Each invocation has polynomial running time; timing is not a separate communication channel. The adversary controls the server and message delivery. All their queries and all replacement-code queries to the KDF are classical.

The firewalls and their local traffic are private. Each new ephemeral publication uses a fresh independent uniform firewall mask. Clients outside U execute the honest protocol, with the state and erasure rules above applied to the values actually used behind their firewalls.

Explainable implementations. Every replacement is functionality maintaining and explainable according to Definition 1. The comparison uses the registered keys and the entire sequence of invocations of that client, on the inputs delivered by its firewall. It includes the local fields of every session and fixes the same random-oracle functions in both executions. Private memory need not follow the honest layout or erasure policy.

Sources and disclosures. The source records and the records of sessions with the same sid use the prescribed session fields. The publication record includes every signed final share forwarded by an honest firewall for a challenger-run session, including a session of a subverted client. Hidden local signing computations are not publications. For a target T with peer P_j , $\text{Src}_W(T)$ contains all such sessions of P_j publishing its accepted peer share before T accepts. Use Src_W in every source clause of the base freshness predicate. Roles, session identifiers and completion status impose no further source restriction.

For $i \notin U$, $\text{RevState}(i, s)$ has the preceding honest-session semantics. For $i \in U$, it returns the current private client memory, including retained data from other sessions, but excluding firewall memory. Every well-formed state query to a slot of such a client records a permanent client-exposure flag $d_i = 1$, even before sampling or after completion, and even if its answer is empty. The usual session reveal flags are recorded as well. Initially all d_i are zero.

For every client, $\text{RevLTK}(i)$ returns only its fixed registered long-term action and signing keys. It does not return other client memory or firewall secrets. A session-key query returns k_i^s , which is $\perp$ before acceptance, and is recorded before its reply is computed. Registration, session limits and the distinct-party Test rule are unchanged.

Freshness. Keep every base freshness clause, using Src_W as specified. In addition, a tested session of P_i with peer P_j is fresh only if

$$\forall h \in \{i, j\} \cap U : d_h = 0$$

at the final guess. Thus no state query to any session of either subverted participant is allowed in a fresh tested exchange, before or after Test. Other clients may still be exposed. Invalid and unfresh tests receive a fair final guess as in the base experiment. Define

$$\text{Adv}_{\Pi_W}^{\text{AKE-FS}}(\mathcal{A}) = \left| \Pr[b = b'] - \frac{1}{2} \right|.$$

Security means that this advantage is negligible for every PPT adversary and admissible choice of replacement code.

Explainability does not constrain retained private data, which accounts for the client-memory restriction in freshness. Long-term-key disclosure after acceptance remains permitted by the other freshness clauses, and honest clients retain the base session/source exposure guarantees.

Definition 20 (AKE Receiver Obliviousness). *An AKE protocol is receiver oblivious (or Signal-conforming) if the initiator can compute the first-message without knowledge of the peer id and long-term public key of the communication peer.*

C Dimension 4 and 8 Isogeny Representations

When one does not have access to a dimension 1 auxiliary isogeny $\psi : E_1 \rightarrow E_3$ of degree b but $b = b_1^2 + b_2^2$ is the sum of two squares, then one considers the dimension 2 isogenies $\Phi : E_1 \times E_1 \rightarrow E_2 \times E_2$ given by $\Phi = \begin{pmatrix} \phi & 0 \\ 0 & \phi \end{pmatrix} = \phi \cdot \text{I}_2$ and $\Psi : E_1 \times E_1 \rightarrow E_1 \times E_1$ given by $\Psi = \begin{pmatrix} b_1 & b_2 \\ -b_2 & b_1 \end{pmatrix}$. As before, we have the following commutative diagram:

$$\begin{array}{ccc} E_1 \times E_1 & \xrightarrow{\Phi} & E_2 \times E_2 \\ \Psi \downarrow & & \downarrow \Psi' \\ E_1 \times E_1 & \xrightarrow{\Phi'} & E_2 \times E_2 \end{array}$$

The dimension 4 isogeny $\Theta : E_2 \times E_2 \times E_1 \times E_1 \rightarrow E_1 \times E_1 \times E_2 \times E_2$ given by $\Theta = \begin{pmatrix} \hat{\Phi} & \hat{\Psi} \\ -\Psi' & \Phi' \end{pmatrix}$, has kernel $\ker \Theta = \{(\Phi(P), \Psi(P)), P \in E_1 \times E_1[N]\}$ and degree $N = a + b$. Since Θ has smooth degree and accessible kernel, it can be evaluated on points $(P, 0, 0, 0) \in E_2 \times E_2 \times E_1 \times E_1$ to obtain $(\hat{\phi}(P), 0, 0, 0)$, allowing to evaluate $\hat{\phi}$ efficiently.

Every integer can be written as the sum of four squares. So, when b is not the sum of two squares, one writes it as a sum of four squares $b = b_1^2 + b_2^2 + b_3^2 + b_4^2$, and define the dimension 4 isogenies $\Phi : E_1^4 \rightarrow E_2^4$

given by $\Phi = \phi \cdot \text{I}_4$ and $\Psi : E_1^4 \rightarrow E_1^4$ given by $\Psi = \begin{pmatrix} b_1 & -b_2 & -b_3 & -b_4 \\ b_2 & b_1 & b_4 & -b_3 \\ b_3 & -b_4 & b_1 & b_2 \\ b_4 & b_3 & -b_2 & b_1 \end{pmatrix}$.

One then proceeds similarly as before, with $\Theta : E_2^4 \times E_1^4 \rightarrow E_1^4 \times E_2^4$ now being a dimension 8 isogeny.