K-Waay: Fast and Deniable Post-Quantum X3DH without Ring Signatures

Daniel Collins and Loïs Huguenin-Dumittan, *EPFL;* Ngoc Khanh Nguyen, *King's College London;* Nicolas Rolin, *Spuerkeess;* Serge Vaudenay, *EPFL*

https://www.usenix.org/conference/usenixsecurity24/presentation/collins

This paper is included in the Proceedings of the 33rd USENIX Security Symposium.

August 14–16, 2024 • Philadelphia, PA, USA

978-1-939133-44-1

Open access to the Proceedings of the 33rd USENIX Security Symposium is sponsored by USENIX.

K-Waay: Fast and Deniable Post-Quantum X3DH without Ring Signatures*

Daniel Collins
EPFL
daniel.collins@epfl.ch

Loïs Huguenin-Dumittan
EPFL
lois.huguenin-dumittan@epfl.ch

Ngoc Khanh Nguyen†
King's College London
nknguyen007@gmail.com

Nicolas Rolin‡
Spuerkeess
nicrolin@hotmail.fr

Serge Vaudenay
EPFL
serge.vaudenay@epfl.ch

Abstract

The Signal protocol and its X3DH key exchange core are regularly used by billions of people in applications like WhatsApp but are unfortunately not quantum-secure. Thus, designing an efficient and post-quantum secure X3DH alternative is paramount. Notably, X3DH supports *asynchronicity*, as parties can immediately derive keys after uploading them to a central server, and *deniability*, allowing parties to plausibly deny having completed key exchange. To satisfy these constraints, existing post-quantum X3DH proposals use ring signatures (or equivalently a form of designated-verifier signatures) to provide authentication without compromising deniability as regular signatures would. Existing ring signature schemes, however, have some drawbacks. Notably, they are not generally proven secure in the quantum random oracle model (QROM) and so the quantum security of parameters that are proposed is unclear and likely weaker than claimed. In addition, they are generally slower than standard primitives like KEMs.

In this work, we propose an efficient, deniable and post-quantum X3DH-like protocol that we call K-Waay, that does *not* rely on ring signatures. At its core, K-Waay uses a split-KEM, a primitive introduced by Brendel et al. [SAC 2020], to provide Diffie-Hellman-like implicit authentication and secrecy guarantees. Along the way, we revisit the formalism of Brendel et al. and identify that additional security properties are required to prove a split-KEM-based protocol secure. We instantiate split-KEM by building a protocol based on the Frodo key exchange protocol relying on the *plain* LWE assumption: our proofs might be of independent interest as we show it satisfies our novel unforgeability and deniability security notions. Finally, we complement our theoretical results by thoroughly benchmarking both K-Waay and existing X3DH protocols. Our results show even when using plain LWE and a conservative choice of parameters that K-Waay is significantly faster than previous work.

*The full version of this paper can be found on ePrint [25].
†All of this work was completed while the author was working at EPFL.
‡Part of this work was completed while the author was studying at EPFL.

1 Introduction

Researchers for several years now have sought to build cryptographic primitives and protocols that are resistant to efficient quantum attacks [58]. This is highly evidenced with the NIST Post-Quantum Cryptography competition for standardising quantum-safe key encapsulation mechanisms (KEM) and signatures, organised by the United States National Institute of Standards and Technology (NIST). Recently, four schemes were selected by NIST for standardisation, out of which three rely on algebraic lattices. Indeed, with the US National Security Agency releasing their new CNSA 2.0 Suite [62], which says that CRYSTALS-Kyber [13] and CRYSTALS-Dilithium [32] should be the main cryptographic force for communication security beginning from 2030, lattices are a natural candidate for building more advanced cryptographic primitives, such as secure messaging.

The widely used Signal protocol for secure messaging as currently deployed is not quantum-safe since it is based on Diffie-Hellman key exchange [30]. The protocol, used in applications like Signal and WhatsApp, comprises two components, namely 1) the X3DH key exchange [50] which is used to bootstrap sessions of 2) the Double Ratchet messaging protocol [54]. The Double Ratchet has been investigated in a line of recent works [4, 10, 18] that each neatly abstract the protocol into primitives like so-called *continuous key agreement*. Fortunately, these primitives have post-quantum (PQ) instantiations that leaves the core structure and resulting security guarantees of the Double Ratchet in place.

In standard X3DH, parties use a mixture of ephemeral (one-time), semi-static (many-time but temporary) and long-term keys. First, parties upload their keying material to a central server or public key infrastructure in a so-called prekey bundle. A party can then derive a session key by downloading their partner's bundle and performing three (or four) Diffie-Hellman key exchanges with a mixture of ephemeral and long-term (resp. plus semi-static) keys, ensuring at least confidentiality if the ephemeral *or* long-term key of each party is corrupted.

Observe that X3DH does not use signatures after signed prekeys are uploaded: at that point, the DH exchanges provide (implicit) authentication guarantees. Consequently, the protocol provides a level of *deniability* [33, 55, 60] as was formalized by Vatandas et al. [63]: informally, a participant can deny having performed key exchange with its counterpart. This is an important privacy guarantee that prevents (at least on a cryptographic level) a conversation transcript from incriminating an unsuspecting party, which is especially pertinent in situations like whistleblowing and protesting.

In May 2023, Signal published code corresponding to their initial *hybrid* post-quantum key exchange solution, namely the "PQXDH protocol".[1] Like in X3DH, several Diffie-Hellman key exchanges are performed at once, but in PQXDH, parties upload prekey bundles that also contain a Kyber-1024 public key that the initiator additionally encapsulates to the responder with. Moreover, prekey bundles are still signed with the same signature scheme as regular X3DH based on Curve25519 [8]. Although PQXDH appears to provide post-quantum confidentiality, which is an important first step towards post-quantum security as it prevents "store-now-decrypt-later" kind of attacks, it does not provide post-quantum *authentication* as an active quantum attacker can trivially forge pre-key bundles. It is thus prudent to design a suitable X3DH alternative that is *fully* post-quantum secure.

A natural direction for building such a protocol is to emulate X3DH's structure by replacing Diffie-Hellman key exchange with a cryptographic group action, such as CSIDH [21]. In order to broadly capture this protocol structure, Brendel et al. [17] introduce a primitive called *split-KEM* that captures the symmetry of e.g. Diffie-Hellman. In a split-KEM, a party A encapsulates to their partner B by using their own secret sk_A and their partner's public key pk_B to produce a ciphertext; B then decapsulates it using sk_B and pk_A. The authors define indistinguishability-based security notions and notice that Frodo [12] lattice-based key-exchange fulfills the split-KEM syntax and the weakest notion of indistinguishability they define.[2] Although they present a X3DH-like protocol, they do not define a security model, and, looking ahead, their split-KEM security notions do not suffice to construct X3DH-like key exchange with authenticity and deniability.

In two recent works, Hashimoto et al. [37, 38] and Brendel et al. [16] concurrently proposed instead to construct X3DH-like key exchange using KEMs directly. Since a core feature of X3DH is its *asynchronicity*, a challenge-response protocol cannot be employed using KEMs alone to provide authentication [57]. Thus to ensure deniability, two seemingly different approaches were proposed: Hashimoto et al. [37] apply ring signatures while Brendel et al. [16] use a flavour of designated verifier signatures; these primitives were later shown to be equivalent [38].

[1] https://signal.org/docs/specifications/pqxdh.

[2] The construction can conceptually be seen as instantiating the lattice-based cryptographic group action from [9].

As described in the aforementioned works, the currently most efficient post-quantum ring signatures [9, 34, 44, 48, 64] are proven to be secure in the random oracle model [7] and can enjoy signatures that are a handful of kilobytes large. Often, however, the constructions do not come with a security proof in the quantum random oracle model (QROM) [11]. In this vein, parameters are generally optimistically chosen as the security loss incurred by proofs in the ROM is not taken into account when setting these, without even mentioning QROM loss, which is usually much larger. Further, security notions can differ between papers, making it less clear exactly when they are appropriate for use.

More generally, it is of interest to determine the cost (or overhead) that deniability incurs in (X3DH-like) key exchange. Towards this goal, Hashimoto et al. [38] provide benchmarks for their baseline, non-deniable X3DH-like protocol based on signatures and KEMs, and Brendel et al. [16] consider parameter sizes for (but do not benchmark) existing ring and designated verifier signatures. As such, a more fine-grained and detailed evaluation will help inform practitioners on the overhead incurred by deniability in the post-quantum setting.

While the use of ring signatures to build PQ and deniable X3DH is at least theoretically understood, this far from exhausts the protocol design space. Motivated by this and the above discussion, we therefore first ask: Can we design a provably-secure, efficient and deniable post-quantum X3DH alternative that does *not* require ring signatures?

1.1 Our Results

In this work, we propose an efficient, deniable and post-quantum X3DH-like protocol without ring signatures that we call K-Waay. To summarise our contributions:

- Towards building our protocol, we revisit the split-KEM formalism proposed by Brendel et al. [17] and deduce that several additional properties, namely notions of authenticity and deniability, are needed to construct a secure X3DH-like deniable authenticated key exchange protocol (DAKE).
- We propose K-Waay, a X3DH-like DAKE that uses deniable and unforgeable split-KEM at its core. Our protocol uses signatures to sign prekeys, and then uses ephemeral KEM, long-term KEM and split-KEM for the final key exchange step.
- The main drawback of a naive version of our protocol is that parties can run out of ephemeral keys, thus making the protocol synchronous if this happens (e.g. Alice needs to wait for Bob's fresh ephemeral key before sending a message). While such a problem would rarely occur in practice, given enough keys are uploaded on the server, we propose a simple trick that makes the

reuse of ephemeral keys possible on the receiver's side for messages they received while offline. We think this trick could be of independent interest as it – perhaps surprisingly – allows for a specific kind of key reuse for a split-KEM that is *not* IND-CCA secure.

- We prove key indistinguishability in our model that captures ephemeral key reuse and session state exposure, and prove a variant of deniability that strengthens the notion of Brendel et al. [16] by additionally leaking the victim's session state to the adversary in the security game.
- We instantiate a post-quantum split-KEM secure under our new security notions derived from the Frodo key exchange protocol (FrodoKEX) [12] based on the plain LWE assumption. We then use a transform in the (Q)ROM to prove it UNF-1KMA (i.e. our new unforgeability definition for split-KEM). This construction incurs a security loss as usual in the (Q)ROM, and we pick conservative parameters such that our split-KEM has 128 (resp. 64) bits of classical (resp. quantum) security if the adversary is limited to $\approx 2^{64}$ queries (resp. $2^{64}/d$ quantum queries) to the random oracle, where d represents the maximal number of distinct participants that can send a message to a given receiver while the latter is offline. In other words, our parameters take into account the loss due to the (Q)ROM proof.
- We benchmark our protocol K-Waay using our modified version of FrodoKEX (which we call FrodoKEX+) as the split-KEM, along with standard X3DH and the two previous proposals for PQ X3DH-like AKE [16, 38]. We find that while K-Waay has larger prekeys, it is 6× faster compared to these. In addition, the only non-standard primitive we use in K-Waay (i.e. FrodoKEX+) is based on both an assumption (i.e. LWE) and a scheme (FrodoKEM) that have been thoroughly scrutinized by the cryptographic community. Overall, we believe our protocol more mature for short to medium-term integration compared to previous work based on ring signatures.

1.2 Technical Overview

1.2.1 X3DH-like key exchange.

A quantum-secure X3DH-like protocol should satisfy certain properties. Apart from satisfying standard authenticated key exchange (AKE) properties like secrecy and authentication, it should also be *asynchronous*. That is, parties should be able to upload keying material to a central server, after which an initiating party can derive a session key immediately with their counterpart who may be offline. This also entails *receiver-obliviousness*, using the language of Hashimoto et al. [38], as the initial key upload should not depend on the keys of any other party. Another is *deniability*, allowing parties to claim that they plausibly did not participate in the key exchange. Note that we cannot possibly ensure that parties can claim that they never uploaded prekeys as they are signed (and using ring signatures, e.g., would violate receiver-obliviousness). Finally, a DAKE should, like X3DH, provides security guarantees even if the session state of a party is leaked.

1.2.2 Revisiting split-KEM.

In an attempt to model the primitive central to X3DH-like AKE, Brendel et al. [17] introduced *split-KEM*, which is similar to a standard KEM except the encapsulator can contribute to the derived key. However, we discovered that the accompanying security definitions were not sufficient to use such a primitive as the main component of a KE. The reason being that their notions ensure that an encapsulated ciphertext will not leak information on its encapsulated key, but not that only the sender can send a "legitimate" ciphertext to the sender (or that only the sender and receiver can derive a common key). In other words, there is no guarantee of implicit authentication. Therefore, we introduce the notion of unforgeability against one known-message attacks for split-KEM (UNF-1KMA), which states that if Bob receives a message allegedly sent by Alice, either Alice really sent it or the decapsulation will fail. Jumping ahead, this will be used in the security proof of the protocol to argue that either the adversary relayed a legit split-KEM ciphertext to the receiver that the adversary cannot learn the decapsulation of, or the sender aborts as the ciphertext is forged.

We also introduce an intermediary notion of decaps-OW-CPA, which says that an adversary should not be able to recover a key *decapsulated* by some party without knowing the sender's or the receiver's secret key. We will prove that our lattice-based split-KEM satisfies such a definition, then we will apply some transform in the (Q)ROM to obtain a UNF-1KMA split-KEM.

Finally, we also define the notion of deniability for split-KEM, which states that no party J can be convinced that a party A sent a given ciphertext to B, even knowing B's secret key but assuming both parties did not deviate from the protocol. This models a setting where B communicates with A and later tries to frame the latter by giving the transcript and their own secret key to J.

1.2.3 Construction.

As any X3DH-like protocol, our construction works in 4 phases: long-term key generation, prekey generation, send and receive. The first observation we make is that in X3DH implementation, prekey bundles are signed with a long-term signing key before being uploaded to the server. This fact is often abstracted away in formal analysis as it hurts the claims one can make about the deniability of X3DH: as a signature is undeniable by definition, users cannot deny they *participated*

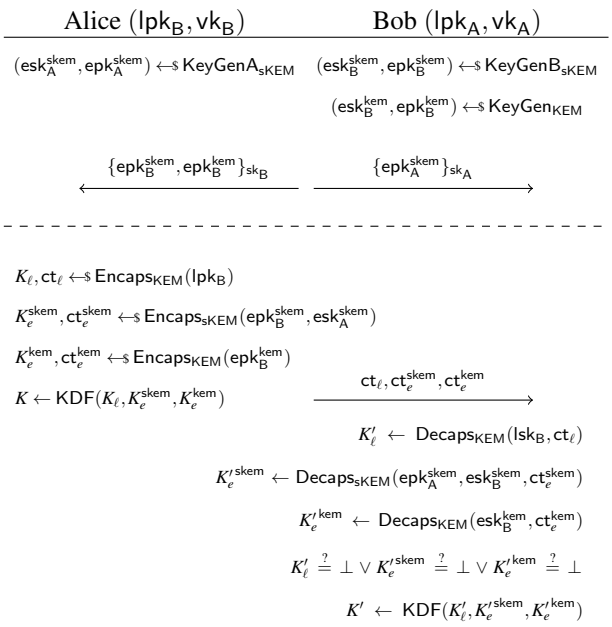

Figure 1: High-level overview of the K-Waay protocol. Values in brackets $\{\cdot\}_{\mathsf{sk}}$ are signed with sk and the signature is verified upon reception. For clarity, we omit the calculation and addition of session identifier sid to KDF.

in the protocol. Based on this, our goal was to achieve some level of peer-deniability [26], where parties can deny they communicated with someone in particular, and to leverage the fact that we use signatures to authenticate the prekeys. Our protocol works then as follows (see Figure 1 for a high-level overview). The long-term key pair consists of a KEM and signature key pair, the latter being used to sign the prekey, which comprises an ephemeral KEM key pair and ephemeral split-KEM key pair. The former is used for forward secrecy while the second is used for the implicit authentication of the sender. Although usually ephemeral keys cannot be used for authentication as they are dynamic, in our case we can since they are authenticated (i.e. signed) by their owner. Then, the sender encapsulates against both KEM public keys of the receiver, and uses their own split-KEM secret key and the receiver's public key to derive a split-KEM ciphertext. Upon decapsulation, the receiver recovers the three keys and combines them using a PRF to derive the shared key.

Ephemeral split-KEM key reuse. The way our protocol is described above works perfectly well if the split-KEM satisfies the UNF-1KMA unforgeability notion introduced above. However, in practice, it could happen that some party (e.g. Bob) is offline for too long and all their ephemeral split-KEM keys have been used. If that occurs, another sender would have to wait for Bob to come online and upload new keys before they can send him a message.

We fix this issue by modifying the protocol as follows: when Bob's ephemeral public keys have run out on the server, a sender can simply reuse one of them. Then, when Bob is back online, he groups the ciphertexts corresponding to the same public-key and decrypts all ciphertexts in a group *at once*. If one or more of the decapsulations in a group fails, Bob outputs $\perp$ for *all* ciphertexts and e.g., restarts the protocol. Otherwise, Bob proceeds as before (and *never* decapsulates again using the same split-KEM key). We formally model this key reuse with an algorithm BatchReceive that takes as input a given session state and one or more messages to be received.

Security. We show this version of the protocol is secure assuming the split-KEM satisfies a stronger notion than IND-CPA that we call IND-1BatchCCA. This definition is the same as traditional IND-CPA (adapted to the split-KEM syntax), except the adversary can query a decapsulation oracle *once* with multiple public keys and ciphertexts, and the oracle returns $\perp$ if *one or more* of the decapsulations failed, and the resulting keys otherwise. We show that one can easily build an IND-1BatchCCA split-KEM out of a CPA secure one in the (Q)ROM, conveniently using the same transform mentioned above that builds a UNF-1KMA scheme out of a decaps-OW-CPA one.

As in previous protocols [16, 38], the long-term KEM provides implicit authentication of the receiver as only they can decrypt. As mentioned above, the ephemeral KEM provides forward secrecy, and the UNF-1KMA/IND-1BatchCCA split-KEM provides implicit authentication of the sender, as it guarantees that only the sender could have sent a ciphertext that correctly decapsulates (unforgeability), and no adversary knows what is inside that ciphertext (indistinguishability), even after seeing the decapsulation of one batch of ciphertexts encapsulated against the same public-key (if all decapsulated correctly). We note that the sender-to-receiver authentication depends both on a long-term key (i.e. the signing key) and an ephemeral one (the split-KEM key). Consequently, our model (that allows session state exposure) is more restrictive than that of Hashimoto et al. [38], since in particular it suffices for the adversary to learn a receiver's ephemeral state during key exchange to forge a message that the receiver accepts. Intuitively, this is because split-KEM is effectively a symmetric primitive. Nevertheless, the security that we achieve is stronger than weak forward security without session state exposure.

Deniable split-KEM from lattices. We provide the first lattice-based split-KEM which satisfies both deniability and UNF-1KMA security. Our starting point is the Frodo key-exchange (FrodoKEX) [12], which was identified (among other schemes) as a split-KEM by Brendel et al. [17], the security of which relies on the well-known Learning with Errors (LWE) problem [56]. We highlight that the vanilla

construction of FrodoKEX does not enjoy the aforementioned properties. Indeed, when looking closely at the security games of deniability and UNF-1KMA, partial information about the secret keys are revealed - thus making a reduction to LWE completely non-trivial[3]. We circumvent this problem in two ways.

First, we reduce deniability of our scheme to a so-called Extended-LWE problem [2], where in addition to a standard LWE instance, the adversary is given a short random combination of the secret coefficients. We show that deniability of our scheme reduces straightforwardly to Extended-LWE, and then follow the methodology of Alperin-Sheriff and Peikert [2] to reduce it further to plain LWE.

Towards UNF-1KMA security, we slightly modify the Frodo split-KEM by introducing masking terms. As the name suggests, they are used to hide the partial information about secret keys. Consequently, this enables us to simulate the decapsulation mechanism via rejection sampling [47]. In Section 5.2 we discuss the necessity of these (seemingly artificial) changes.

1.3 Additional Related Work

The security of X3DH has been modelled in detail by Cohn-Gordon et al. [22]. Vatandas et al. [63] investigate the deniability of X3DH and similar key exchange protocols under the deniability notion of Di Raimondo et al. [55], requiring strong knowledge-of-exponent-type assumptions to prove X3DH secure. Dobson and Galbraith [31] propose a SIDH-based X3DH-like protocol which is unfortunately now broken [20]. Very recently, [40] prove X3DH tightly-secure in the generic group model under a new multi-user assumption although do not allow the adversary to expose parties' session states.

Unger and Goldberg build a number of different DAKEs [60, 61]. However, the protocols do not provide post-quantum guarantees: only in their later paper [61] is it suggested to add a PQ KEM for post-quantum *confidentiality* and the authors do not propose a more comprehensive hybrid protocol. Nevertheless, the protocols provide relatively strong online deniability (i.e. where a judge and a party can communicate while trying to frame another party) at the expense of stronger primitives like dual-receiver encryption and non-committing encryption.

Alwen et al. [3] define the notion of authenticated key encapsulation mechanism (AKEM) and some security definitions. AKEM captures the same primitive as a split-KEM, but we opted for the syntax and language of the latter as it was meant to be used in a X3DH-like protocol.

Cremers and Feltz [26] introduce peer deniability, which captures the kind of participation deniability property we are after in this paper, namely that a party cannot deny using a system but can deny communicating with a particular party. However, their security notion does not require the simulator to output the session key and the adversary to distinguish between the real and simulated key, and so composability issues may arise from using it.

[3] Nevertheless, we found no deniability/UNF-1KMA attack on [17].

2 Preliminaries

In this work, we denote by *efficient* adversary a probabilistic polynomial-time or quantum polynomial-time algorithm unless otherwise specified. Let $[n] = \{1, \ldots, n\}$, i.e. the set of integers between 1 and n.

2.1 Triple PRF

We first define the notion of a triple PRF, a natural generalisation of a dual PRF:

Definition 2.1 (Triple PRF)**.** *Let $F : \mathcal{K} \times \mathcal{K} \times \mathcal{K} \times D \to R$ be a function. We consider the game shown in Figure 2. We say that F is* 3PRF *if for all efficient adversaries, we have:*

$$\mathsf{Adv}^{\mathrm{3prf}}_{F}(\mathcal{A}) := \max_{i \in \{1,2,3\}} \left| \Pr\left[\mathrm{PRF}_{F_i}(\mathcal{A}) \Rightarrow 1\right] - \frac{1}{2} \right| = \mathsf{negl}\,.$$

where F_i denotes F keyed in its i-th argument for $i \in \{1,2,3\}$.

Note that the notion of a triple PRF generalises the now common notion of a *dual PRF* [6]. A triple PRF F_{triple} can be trivially constructed in the random oracle model, or from a dual PRF F_{dual} as $F_{triple}(k_1,k_2,k_3,x) = F_{dual}(k_1, F_{dual}(k_2,k_3,x),x)$.

$\mathrm{PRF}_F(\mathcal{A})$	$\mathrm{PRF}(a,b,c)$
Sample random function G	**if** $b = 0$
$k \leftarrow\$ \mathcal{K}$	**return** $F_k(a,b,c)$
$b \leftarrow\$ \{0,1\}$	**else**
$b' \leftarrow\$ \mathcal{A}^{\mathrm{PRF}}(1^\lambda)$	**return** $G(a,b,c)$
return $1_{b'=b}$	

Figure 2: PRF game for function F_k taking three arguments as input.

2.2 Split-KEM

We next define split-KEM, which was introduced by Brendel et al. [17].

Definition 2.2 (Split-KEM)**.** *A split-KEM* sKEM *is a tuple of four efficient algorithms* $(\mathsf{KeyGenA}, \mathsf{KeyGenB}, \mathsf{Encaps}, \mathsf{Decaps})$ *defined as follows:*

- $(\mathsf{pk_A}, \mathsf{sk_A}) \leftarrow\$ \mathsf{KeyGenA}(1^\lambda)$ *(resp.* $(\mathsf{pk_B}, \mathsf{sk_B}) \leftarrow\$ \mathsf{KeyGenB}(1^\lambda)$*): The key generation function of the first/second party takes the security parameter λ as input, and outputs a pair of public/secret keys* $(\mathsf{pk_A}, \mathsf{sk_A})$ *(resp.* $(\mathsf{pk_B}, \mathsf{sk_B})$*).*

- $K, \mathsf{ct} \leftarrow\$ \mathsf{Encaps}(\mathsf{pk}_{\mathcal{P}}, \mathsf{sk}_{\overline{\mathcal{P}}})$*: The encapsulation function takes the public key* $\mathsf{pk}_{\mathcal{P}}$ *of a party* $\mathcal{P} \in \{\mathsf{A}, \mathsf{B}\}$ *and the other party's secret key* $\mathsf{sk}_{\overline{\mathcal{P}}}$ *as inputs, and outputs a ciphertext* ct *and a key* K*.*
- $K/\perp \leftarrow \mathsf{Decaps}(\mathsf{pk}_{\overline{\mathcal{P}}}, \mathsf{sk}_{\mathcal{P}}, \mathsf{ct})$*: The decapsulation function takes the secret key* $\mathsf{sk}_{\mathcal{P}}$ *of a party* $\mathcal{P} \in \{\mathsf{A}, \mathsf{B}\}$*, the other party's public key* $\mathsf{pk}_{\overline{\mathcal{P}}}$ *and a ciphertext* ct *as inputs, and outputs a key* K *or the error symbol* $\perp$*.*

Finally, we say a split-KEM is $(1-\delta)$*-correct if*

$$\Pr\left[K \neq K' : \begin{array}{c}(\mathsf{pk}_\mathsf{A}, \mathsf{sk}_\mathsf{A}) \leftarrow\$ \mathsf{KeyGenA}(1^\lambda);\\ (\mathsf{pk}_\mathsf{B}, \mathsf{sk}_\mathsf{B}) \leftarrow\$ \mathsf{KeyGenB}(1^\lambda);\\ K, \mathsf{ct} \leftarrow\$ \mathsf{Encaps}(\mathsf{pk}_\mathsf{B}, \mathsf{sk}_\mathsf{A});\\ K' \leftarrow \mathsf{Decaps}(\mathsf{pk}_\mathsf{A}, \mathsf{sk}_\mathsf{B}, \mathsf{ct})\end{array}\right] \leq \delta\,.$$

Intuitively, a split-KEM is similar to a normal KEM except material from both participants is used for encapsulation (i.e. the final key will depend on both parties' secret/public keys). In a X3DH-like protocol, it can be used to implicitly authenticate the party encapsulating. For security, we first define one-wayness (OW-CPA) for sKEM, which is very similar to the usual one for KEM and another new notion called IND-1BatchCCA. Looking ahead, we will show that any OW-CPA split-KEM can easily be transformed into a IND-1BatchCCA one.

Definition 2.3 (split-KEM OW-CPA)**.** *We consider the* OW-CPA *game defined in Fig. 3. A split-KEM scheme* $\mathsf{sKEM} = (\mathsf{KeyGen}_\mathsf{A}, \mathsf{KeyGenB}, \mathsf{Encaps}, \mathsf{Decaps})$ *is* OW-CPA *if for any efficient adversary* $\mathcal{A}$ *we have*

$$\mathsf{Adv}^{\text{ow-cpa}}_{\mathsf{sKEM}}(\mathcal{A}) = \Pr\left[\text{OW-CPA}_{\mathsf{sKEM}}(\mathcal{A}) \Rightarrow 1\right] = \mathsf{negl}\,.$$

Definition 2.4 (split-KEM IND-1BatchCCA)**.** *We consider the* IND-1BatchCCA *game defined in Fig. 3. Let* $\mathcal{K}$ *be a finite key space. A split-KEM scheme over* $\mathcal{K}$ $\mathsf{sKEM} = (\mathsf{KeyGen}_\mathsf{A}, \mathsf{KeyGenB}, \mathsf{Encaps}, \mathsf{Decaps})$ *is* IND-1BatchCCA *if for any efficient adversary* $\mathcal{A}$ *we have*

$$\mathsf{Adv}^{\text{ind-1batchcca}}_{\mathsf{sKEM}}(\mathcal{A}) := \left|\Pr\left[\text{IND-1BatchCCA}_{\mathsf{sKEM}}(\mathcal{A}) \Rightarrow 1\right] - \tfrac{1}{2}\right| = \mathsf{negl}\,.$$

On the original split-KEM security. In the original work on split-KEM by Brendel et al. [17], the authors defined several indistinguishability notions meant to capture which level of security a split-KEM should achieve in order to be used as part of a X3DH-like key exchange. This goes from nn-IND-CCA security, which is the same as IND-CPA security for split-KEM, to mm-IND-CCA security (equivalent to some kind of IND-CCA security). Many intermediary notions of the form xy-IND-CCA, $\mathrm{x}, \mathrm{y} \in \{\mathrm{n}, \mathrm{s}, \mathrm{m}\}$ were also defined, where x (resp. y) specifies the number of queries an adversary can make to the decapsulation (resp. encapsulation) oracle (i.e. none, single, or many). We refer the reader to the original work on split-KEM [17] or the full version of this paper [25] for the formal definitions.

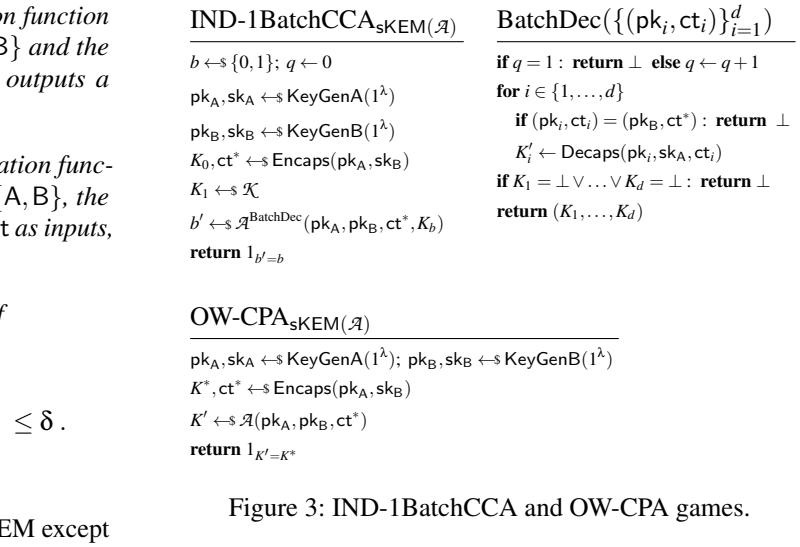

Figure 3: IND-1BatchCCA and OW-CPA games.

We recall that the advantage of split-KEMs over normal KEMs is that they capture the fact that the party encapsulating can contribute towards the shared key, whereas it is not the case with KEMs, as the encapsulation function only takes the receiving party's public key as input. In particular, this means that KEMs cannot be used for implicit authentication of the encapsulator, unlike split-KEMs. However, we argue that the original xy-IND-CCA definitions for split-KEMs [17] do not capture implicit authentication either and thus are not suited for their purpose (i.e. building an asynchronous DAKE). In fact, any IND-CPA (resp. IND-CCA) KEM can be seen as an (asymmetric) split-KEM satisfying IND-CPA (resp. mm-IND-CCA), where the sender's pair of key is set to $(\perp, \perp)$ and encapsulation is done using only the receiver's public key.

More formally, imagine a setting where Alice and Bob know each other's public key, and Alice wants to implicitly authenticate to Bob using a split-KEM. In addition, we assume a mm-IND-CCA split-KEM sKEM_0 exists (note mm-IND-CCA security is the strongest so this holds for all weaker notions). We first modify sKEM_0 s.t. on a special ciphertext $\mathsf{ct}^\star$ not in the original ciphertext space, Decaps returns a constant key $K^\star$. Let's call this modified scheme sKEM. We observe that sKEM is still mm-IND-CCA secure as no adversary can break an honestly-generated challenge ciphertext. Now, implicit authentication means that if Bob decapsulates a ciphertext and obtains a key K, then only Alice knows K. However, in our case, any adversary can send $\mathsf{ct}^\star$ to Bob and set their own key to $K^\star$. Both the adversary and Bob will share the same key and implicit authentication does not hold. In a way, xy-IND-CCA security does not prevent forgery.

This leads us to define our notion of UNF-1KMA security for split-KEMs below which, along with OW-CPA (which can be turned into IND-1BatchCCA), guarantees that only

$\underline{\text{UNF-1KMA}_{\mathsf{sKEM}}(\mathcal{A})}$

$\mathsf{pk_A}, \mathsf{sk_A} \leftarrow\!\$\ \mathsf{KeyGenA}(1^\lambda)$
$\mathsf{pk_B}, \mathsf{sk_B} \leftarrow\!\$\ \mathsf{KeyGenB}(1^\lambda)$
$K_\mathsf{B}, \mathsf{ct} \leftarrow\!\$\ \mathsf{Encaps}(\mathsf{pk_A}, \mathsf{sk_B})$
$\mathsf{ct}' \leftarrow\!\$\ \mathcal{A}(\mathsf{pk_A}, \mathsf{pk_B}, \mathsf{ct}, K_\mathsf{B})$
if $\mathsf{ct}' = \mathsf{ct}$: **return** 0
$K_\mathsf{A} \leftarrow \mathsf{Decaps}(\mathsf{pk_B}, \mathsf{sk_A}, \mathsf{ct}')$
if $K_\mathsf{A} = \perp$: **return** 0
return 1

$\underline{\text{decaps-OW-CPA}_{\mathsf{sKEM}}(\mathcal{A})}$

$b \leftarrow\!\$\ \{0,1\}$
$\mathsf{pk_A}, \mathsf{sk_A} \leftarrow\!\$\ \mathsf{KeyGenA}(1^\lambda)$
$\mathsf{pk_B}, \mathsf{sk_B} \leftarrow\!\$\ \mathsf{KeyGenB}(1^\lambda)$
$K_\mathsf{B}, \mathsf{ct} \leftarrow\!\$\ \mathsf{Encaps}(\mathsf{pk_A}, \mathsf{sk_B})$
$K'_\mathsf{A}, \mathsf{ct}' \leftarrow\!\$\ \mathcal{A}(\mathsf{pk_A}, \mathsf{pk_B}, \mathsf{ct})$
$K_\mathsf{A} \leftarrow \mathsf{Decaps}(\mathsf{pk_B}, \mathsf{sk_A}, \mathsf{ct}')$
if $K_\mathsf{A} = \perp$: **return** 0
return $1_{K_\mathsf{A} = K'_\mathsf{A}}$

Figure 4: Games UNF-1KMA and decaps-OW-CPA.

$\underline{\text{DENY}^{\mathsf{REAL}}_{\mathsf{sKEM},\mathsf{Sim}}(\mathcal{A})}$

$(\mathsf{pk_A}, \mathsf{sk_A}) \leftarrow\!\$\ \mathsf{KeyGenA}(1^\lambda)$
$(\mathsf{pk_B}, \mathsf{sk_B}) \leftarrow\!\$\ \mathsf{KeyGenB}(1^\lambda)$
$K, \mathsf{ct} \leftarrow\!\$\ \mathsf{Encaps}(\mathsf{pk_A}, \mathsf{sk_B})$
$b \leftarrow\!\$\ \mathcal{A}(\mathsf{pk_A}, \mathsf{pk_B}, \mathsf{sk_A}, K, \mathsf{ct})$
return b

$\underline{\text{DENY}^{\mathsf{SIM}}_{\mathsf{sKEM},\mathsf{Sim}}(\mathcal{A})}$

$(\mathsf{pk_A}, \mathsf{sk_A}) \leftarrow\!\$\ \mathsf{KeyGenA}(1^\lambda)$
$(\mathsf{pk_B}, \mathsf{sk_B}) \leftarrow\!\$\ \mathsf{KeyGenB}(1^\lambda)$
$K, \mathsf{ct} \leftarrow\!\$\ \mathsf{Sim}(\mathsf{pk_B}, \mathsf{sk_A})$
$b \leftarrow\!\$\ \mathcal{A}(\mathsf{pk_A}, \mathsf{pk_B}, \mathsf{sk_A}, K, \mathsf{ct})$
return b

Figure 5: Deniability game (we assume w.l.o.g. that B encapsulates and A simulates).

Alice (and obviously Bob) can know the result of Bob's decapsulation on some ciphertext. More precisely, UNF-1KMA ensures that no adversary can forge a valid split-KEM ciphertext when knowing a (distinct) legitimate one. We also define a security notion called decaps-OW-CPA that will serve as a building block to build UNF-1KMA. This notion ensures that it is hard for an adversary knowing one legitimate ciphertext ct to come up with a ciphertext ct' (possibly equal to ct) and a key K' s.t. the decapsulation of ct' returns K'.

Definition 2.5 (split-KEM UNF-1KMA)**.** *We consider the* UNF-1KMA *game defined in Fig. 4. A split-KEM scheme* $\mathsf{sKEM} = (\mathsf{KeyGen_A}, \mathsf{KeyGenB}, \mathsf{Encaps}, \mathsf{Decaps})$ *is* UNF-1KMA *if for any efficient adversary* $\mathcal{A}$ *we have*

$$\mathsf{Adv}^{\text{unf-1kma}}_{\mathsf{sKEM}}(\mathcal{A}) := \Pr\left[\text{UNF-1KMA}_{\mathsf{sKEM}}(\mathcal{A}) \Rightarrow 1\right] = \mathsf{negl}\ .$$

Definition 2.6 (split-KEM decaps-OW-CPA)**.** *We consider the* decaps-OW-CPA *game defined in Fig. 4. A split-KEM scheme* $\mathsf{sKEM} = (\mathsf{KeyGen_A}, \mathsf{KeyGenB}, \mathsf{Encaps}, \mathsf{Decaps})$ *is* decaps-OW-CPA *if for any efficient adversary* $\mathcal{A}$ *we have*

$$\mathsf{Adv}^{\text{decaps-ow-cpa}}_{\mathsf{sKEM}}(\mathcal{A}) := \left|\Pr\left[\text{decaps-OW-CPA}_{\mathsf{sKEM}}(\mathcal{A}) \Rightarrow 1\right] - \tfrac{1}{2}\right| = \mathsf{negl}\ .$$

2.3 Deniability

We state here the notion of split-KEM deniability we would like to achieve.

Definition 2.7 (Deniability)**.** *We consider the game shown in Figure 5. We say a split-KEM* sKEM *is* DENY *if there exists a simulator* Sim *s.t. for all efficient adversaries* $\mathcal{A}$*, we have*

$$\mathsf{Adv}^{\text{deny}}_{\mathsf{sKEM},\mathsf{Sim}}(\mathcal{A}) := \Big|\Pr\left[\text{DENY}^{\mathsf{REAL}}_{\mathsf{sKEM},\mathsf{Sim}}(\mathcal{A}) \Rightarrow 1\right] - \Pr\left[\text{DENY}^{\mathsf{SIM}}_{\mathsf{sKEM},\mathsf{Sim}}(\mathcal{A}) \Rightarrow 1\right]\Big| = \mathsf{negl}\ .$$

Note that for the sake of simplicity, we define deniability with respect to B *encapsulating. The other case (i.e.* A *encapsulates) follows by symmetry.*

Informally, the setting considered is the following. Alice and Bob use the split-KEM to establish a shared key (we assume the public keys are only used for this one exchange), and Alice (while following the protocol) wants to frame Bob and prove that he did communicate with her. Therefore, after receiving Bob's ciphertext and deriving the key, Alice gives both public keys, the derived key, the ciphertext and her own secret key to a judge (i.e. the adversary) that must decide whether Bob actually sent the ciphertext that was used to derive the key or not. The scheme is deniable if there is a simulator that, given Alice's view, outputs a ciphertext and a key indistinguishable from the ones output by Bob.

2.4 Model for DAKE

We next describe our model for deniable authenticated key exchange (DAKE) that we tailor to the semantics and flow of X3DH.

Syntax. A DAKE DAKE is a tuple of four efficient algorithms $(\mathsf{KeyGen}, \mathsf{Init}, \mathsf{Send}, \mathsf{BatchReceive})$ defined as follows:

- $(\mathsf{pk}, \mathsf{sk}) \leftarrow\!\$\ \mathsf{KeyGen}(1^\lambda)$. This function takes as input the security parameter λ and outputs the long-term public/secret key pair of the caller.
- $(\mathsf{st}_i, \mathsf{prek}_i) \leftarrow\!\$\ \mathsf{Init}(\mathsf{sk}_i, \mathsf{role})$. This function takes as input a long-term secret key sk_i and a role $\mathsf{role} \in \{\mathsf{sender}, \mathsf{receiver}\}$ and outputs a session state st_i and a prekey bundle prek_i. This function models the creation of key material that will be uploaded to the public key infrastructure by both parties (e.g. the prekey bundle in X3DH). The output values depend only on the public key of party i executing the function.
- $(\mathsf{k}, m) \leftarrow\!\$\ \mathsf{Send}(\mathsf{sk}_i, \mathsf{pk}_j, \mathsf{st}_i, \mathsf{prek}_j)$. This function takes as inputs the secret key of the executing party i, the public key of the intended recipient pk_j, party i's session state st_i and the (claimed) prekey bundle of the intended recipient prek_j, and outputs a key k and a message m.
- $\{k_s\}_s \leftarrow\!\$\ \mathsf{BatchReceive}(\mathsf{sk}_i, \mathsf{st}_i, \{\mathsf{pk}_j, \mathsf{prek}_j, m_j\}_j)$. This function takes as inputs the secret key of the executing party i, an ephemeral state of party i st_i and a vector of size $d \geq 1$ of the form $(\mathsf{pk}_j, \mathsf{prek}_j, m_j)$ for party i's session with the public key of the (claimed) sender pk_j,

party i's session state st_i, the (claimed) prekey bundle of party j prek_j and a message m_j, and outputs a vector of d keys $(\mathsf{k}_1,\ldots,\mathsf{k}_d)$, some or all of which can be $\perp$.

Init explicitly captures parties uploading ephemeral keys to a central server in the first protocol step. This contrasts with the formal modelling in some previous works on X3DH-like key exchange [16, 38] that formally model a three-move key exchange with a single initiator. As Init is independent of keying material from the caller's counterpart, our definition captures so-called *receiver obliviousness* [38] (sometimes *post-specified peers* [19]), corresponding to some, but not all, key exchange protocols in the literature.

The most novel part of our primitive is BatchReceive which in particular captures ephemeral key reuse when uploaded ephemeral keys are exhausted. In the case of key exhaustion, when a party comes back online, they execute BatchReceive several times, where the number of inputs of the form $(\mathsf{pk}_j,\mathsf{prek}_j,m_j)$ in a given BatchReceive call corresponds to how many times ephemeral state st_i is re-used. Otherwise, BatchReceive can be used as one would expect with a single value $(\mathsf{pk}_j,\mathsf{prek}_j,m_j)$ as input.

Security. Due to space limitations and its similarity with previous work [16, 38], we defer a formal description of our key indistinguishability game and some discussion to the full version [25]. We nevertheless define security and discuss its core features below.

Definition 2.8 (DAKE key indistinguishability)**.** *We consider the KIND game described below. We say a DAKE* DAKE *is* KIND *if for all efficient adversaries $\mathcal{A}$ and polynomially-bounded n (the total number of parties), we have*

$$\mathsf{Adv}^{\mathrm{kind}}_{\mathsf{DAKE},n}(\mathcal{A}) := \left|\Pr[\mathrm{KIND}^{n}_{\mathsf{DAKE}}(\mathcal{A}) \Rightarrow 1] - \frac{1}{2}\right| = \mathsf{negl}\,.$$

The KIND game proceeds in two phases in which the adversary drives the execution of several parties, each of which can be executing several concurrent instances of the DAKE. In the first phase, the adversary $\mathcal{A}$ is given access to several oracles, including EXEC (driving protocol execution and including message injection), LTK (for long-term key reveals), REGISTER (for adversarial public key registration), STATE (for ephemeral state exposure) and KEY (for session key exposure). At the end of the first phase, $\mathcal{A}$ makes a TEST query with respect to some session key output by either the sender or receiver, which outputs either the correct session key or a key uniformly sampled by the challenger. In the second phase, the adversary continues to make oracle queries, and finally outputs a guess as to whether the key is real or random.

Apart from the fact we make several extensions to typical AKE modelling to capture BatchReceive, the game is closest to that of Hashimoto et al. [38] except that we additionally enforce correctness checks as done by Brendel et al. [16]. To capture partnering, we consider partner and key identifiers that may be *vectors* for a receiver, such that several sender sessions may be partnered with a receiver session if, for a given sender session, it partners with a part or component of the receiver session. We do not capture semi-static keys explicitly as Brendel et al. [16] do, although in principle they could be captured in Init. We note in particular that our model captures session state exposure, unlike that of Brendel et al. Note that for BatchReceive which can output several keys, just one of the output keys are tested.

$\mathrm{DENY}^{exp}_{\mathsf{DAKE},n,\mathsf{Sim}}(\mathcal{A})$

$b \leftarrow\$ \{0,1\}$
$L \leftarrow \emptyset$
for $i \in [n]$
  $(\mathsf{pk}_i,\mathsf{sk}_i) \leftarrow\$ \mathsf{KeyGen}(1^\lambda)$
  $L \leftarrow L \cup \{(\mathsf{pk}_i,\mathsf{sk}_i)\}$
$b' \leftarrow\$ \mathcal{A}^{\mathrm{CHAL}}(L)$
return $1_{b'=b}$

$\mathrm{CHAL}(s,r)$

require $s \in [n] \wedge r \in [n]$
$(k,m) \leftarrow (\perp,\perp)$
$(\mathsf{st}_s,\mathsf{prek}_s) \leftarrow\$ \mathsf{Init}(\mathsf{sk}_s,\mathsf{sender})$
$(\mathsf{st}_r,\mathsf{prek}_r) \leftarrow\$ \mathsf{Init}(\mathsf{sk}_r,\mathsf{receiver})$
if $b=0$: $(\mathsf{k},m) \leftarrow\$ \mathsf{Send}(\mathsf{sk}_s,\mathsf{pk}_r,\mathsf{st}_s,\mathsf{prek}_r)$
else : $(\mathsf{k},m) \leftarrow\$ \mathsf{Sim}(\mathsf{sk}_r,\mathsf{pk}_s,\mathsf{st}_r,\mathsf{prek}_s,\mathsf{prek}_r)$
$T \leftarrow (\mathsf{prek}_s,\mathsf{prek}_r,m)$
if $exp = \mathsf{true}$: **return** $(\mathsf{k},T,\mathsf{st}_r)$
else return (k,T)

Figure 6: Deniability game.

Trivial attacks. We restrict the adversary's behaviour to prevent 'trivial' attacks (e.g. directly revealing the challenge key) by defining *freshness* predicates. Due to our protocol's design, our notion restricts more than the full forward security notion under session state exposure defined by Hashimoto et al. [38]. Our freshness predicates imply weak forward secrecy and implicit authentication given session state exposure is not allowed (enforced in some recent works [5, 23]). Brendel et al.'s model provide these guarantees but additionally protect against randomness exposure [16]: the standard NAXOS trick [42] can be applied to our protocol to mitigate this attack vector.

2.5 Deniability

We next introduce our security notion for a deniable DAKE. To this end, we introduce security game $\mathrm{DENY}^{exp}_{\mathsf{DAKE},\mathsf{Sim}}$ in Figure 6.

Definition 2.9 (DAKE deniability)**.** *We consider the game shown in Figure 6. We say a DAKE* DAKE *is* DENY^{exp} *for $exp \in \{\mathsf{true},\mathsf{false}\}$ if there exists an efficient simulator* Sim *s.t. for all efficient adversaries $\mathcal{A}$ and polynomially-bounded n, we have*

$$\mathsf{Adv}^{\mathrm{deny}}_{\mathsf{DAKE},\mathsf{Sim},\mathsf{exp}}(\mathcal{A}) := \left|\Pr[\mathrm{DENY}^{exp}_{\mathsf{DAKE},n,\mathsf{Sim}}(\mathcal{A}) \Rightarrow 1] - \frac{1}{2}\right| = \mathsf{negl}\,.$$

Our definition captures the following deniability property. Initially, the judge $\mathcal{A}$ is given the long-term keys of all parties.

$\mathcal{A}$ then observes honest protocol runs between pairs of parties (via CHAL). Depending on the challenge bit b, either Send or a simulator Sim that takes as input the secret keying material of the receiver trying to frame the sender is executed in each run. Moreover, $\mathcal{A}$ is given the prekey messages independent of b and, if the parameter exp is set to true, also the session state of the receiver in each protocol run. The goal of the adversary is to distinguish whether Send or Sim is being called.

Our notion DENY$^{\mathsf{false}}$ corresponds most closely with that of Brendel et al. [16] which was also adopted by Cremers and Zhao [27]. Due to their AKE syntax, they also consider semi-static key pairs which are also given to the adversary. DENY$^{\mathsf{true}}$ provides stronger deniability, corresponding in practice to a receiver who co-operates with a judge by handing over the entire contents of e.g. their phone. Although incomparable formally, our DAKE would not be considered deniable under a notion like that of Brendel et al. [16] since their protocols do not formally model long-term signatures. Finally, note that our definition, like Brendel et al.'s, can be straightforwardly converted to a 'simulation-based' notion like Definition 2.7.

3 K-Waay: Post-Quantum X3DH from Split-KEM

We present our DAKE K-Waay (Key-exchange With asynchrony, authentication and peer-deniability) in Figure 7.

Each party is associated with a long-term public/secret key pair which in K-Waay comprises of a signature and KEM key pair generated in KeyGen. In Init, ephemeral KEM and split-KEM keys for both parties are generated and the public keys are signed with the long-term signature key.

After initialisation, the sender P_i (sometimes initiator) invokes Send that takes the prekey prek_j output by the receiver P_j's Init call as input. After verifying the signature in prek_j, P_i encapsulates to 1) the long-term KEM key of P_j; 2) the ephemeral KEM key contained in prek_j; and 3) the ephemeral split-KEM key contained in prek_j. Note that the split-KEM provides implicit authentication (without it, Send could be simulated without secrets). P_i then combines the keys using a KDF and outputs the key and its message for P_j consisting of the three encapsulation ciphertexts. Receiving is analogous: receiver P_i verifies P_j's prekey, decapsulates using its three respective secret keys and derives the session key. If P_i's prekeys have run out, it is possible that multiple P_j's have sent using the same prekey prek_i. In that case, P_i decapsulates for all sessions using the same secret keys but aborts if *any* split-KEM decapsulation failed in *any* of the session.

3.1 Security

Theorem 1. *Consider* $(1-\delta_{\mathsf{EKEM}})$*-correct* IND-CCA KEM EKEM, $(1-\delta_{\mathsf{LKEM}})$*-correct* IND-CCA *KEM* LKEM, $(1-\delta_{\mathsf{Sig}})$*-correct* SUF-CMA *signature scheme* Sig *and* $(1-\delta_{\mathsf{sKEM}})$*-correct* IND-1BatchCCA, UNF-1KMA *split-KEM* sKEM *and* 3PRF KDF *used to build K-Waay (Figure 7). Then, we have that for every efficient adversary* $\mathcal{A}$ *that makes at most* q *oracle queries, one can build an adversary* $\mathcal{B}$ *such that*

$$\mathsf{Adv}^{\mathsf{kind}}_{\mathsf{K\text{-}Waay}}(\mathcal{A}) \leq \frac{q}{4}\cdot\left(\delta_{\mathsf{Sig}}+\delta_{\mathsf{LKEM}}+\delta_{\mathsf{EKEM}}+\delta_{\mathsf{sKEM}}\right) + q^2\cdot\left(\varepsilon_{\mathsf{EKEM}}+\varepsilon_{\mathsf{LKEM}}+2\varepsilon_{\mathsf{KDF}}+2\varepsilon_{\mathsf{Sig}}\right) + q^3\cdot\left(\varepsilon_{\mathsf{EKEM}}+\varepsilon_{\mathsf{LKEM}}+\varepsilon_{\mathsf{sKEM}}+3\varepsilon_{\mathsf{KDF}}\right),$$

where $\varepsilon_{\mathsf{EKEM}} = \mathsf{Adv}^{\text{ind-cca}}_{\mathsf{EKEM}}(\mathcal{B})$, $\varepsilon_{\mathsf{LKEM}} = \mathsf{Adv}^{\text{ind-cca}}_{\mathsf{LKEM}}(\mathcal{B})$, $\varepsilon_{\mathsf{Sig}} = \mathsf{Adv}^{\text{suf-cma}}_{\mathsf{Sig}}(\mathcal{B})$, $\varepsilon_{\mathsf{sKEM}} = \mathsf{Adv}^{\text{ind-1batchcca}}_{\mathsf{sKEM}}(\mathcal{B}) + \mathsf{Adv}^{\text{unf-1kma}}_{\mathsf{sKEM}}(\mathcal{B})$ *and* $\varepsilon_{\mathsf{KDF}} = \mathsf{Adv}^{\text{3prf}}_{\mathsf{KDF}}(\mathcal{B})$.

Proof. We defer the proof to the full version [25]. At a high level, we first argue protocol correctness by invoking the correctness of the underlying primitives. Then, we partition the adversary's behaviour into three classes, depending on whether 1) the test session is partnered; 2) it is unpartnered

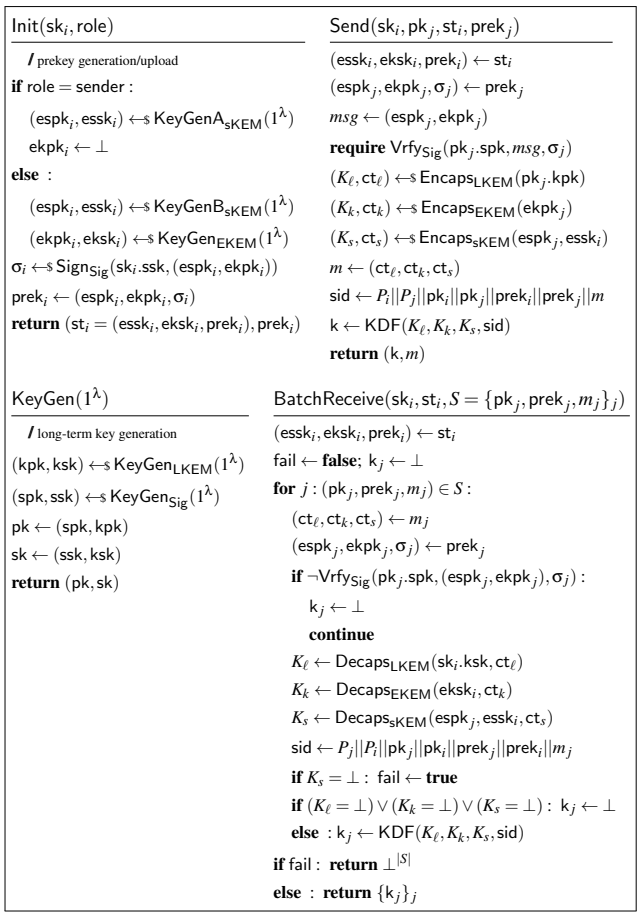

Figure 7: K-Waay: X3DH-like DAKE from IND-CCA KEMs EKEM and LKEM, SUF-CMA signature scheme Sig and IND-1BatchCCA and UNF-1KMA split-KEM sKEM.

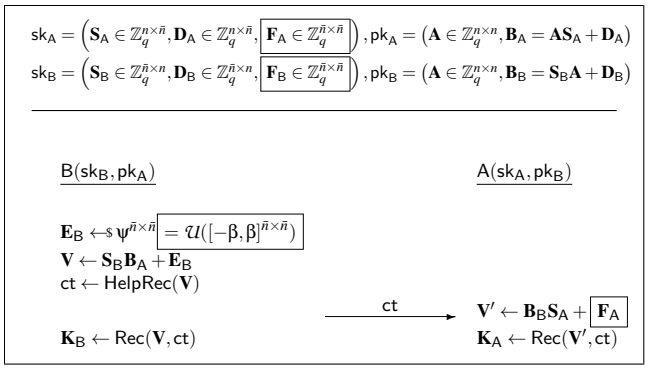

Figure 8: The modified FrodoKEX [12] as a split-KEM. Boxed terms correspond to the changes made to the original construction [12]. Here, Rec and HelpRec are the reconciliation mechanisms [53] used to derive the same key (see the full version [25] for details).

and the tester is a sender and 3) it is a receiver, and, depending on the combination of corruptions/reveals the adversary makes, reduce to the security of a primitive that the adversary does not have access to the secrets of. □

Theorem 2. *Consider deniable split-KEM* sKEM *with simulator* $\mathsf{Sim}_{\mathsf{sKEM}}$ *used to build K-Waay (Figure 7). Then, we have that for every efficient adversary* $\mathcal{A}$ *that makes at most* q *oracle queries, there exists an efficient* Sim *s.t. one can build an adversary* $\mathcal{B}$ *such that for* $\mathsf{exp} \in \{\mathsf{true}, \mathsf{false}\}$ *we have:*

$$\mathsf{Adv}^{\mathsf{deny}}_{\mathsf{K\text{-}Waay},\mathsf{Sim},\mathsf{exp}}(\mathcal{A}) \leq q \cdot \mathsf{Adv}^{\mathsf{deny}}_{\mathsf{sKEM},\mathsf{Sim}_{\mathsf{sKEM}}}(\mathcal{B}) \ .$$

Proof. We defer the proof to the full version [25], noting we construct Sim using split-KEM simulator $\mathsf{Sim}_{\mathsf{sKEM}}$ then apply a standard hybrid argument that replaces Send calls in CHAL queries with calls to $\mathsf{Sim}_{\mathsf{sKEM}}$. □

4 Deniable Split-KEM from Lattices

In this section, we give a technical overview of our new lattice-based split-KEM. More formal treatment of the construction can be found in the full version [25].

4.1 Overview

We start with the Frodo key-exchange by Bos et al. [12], which corresponds to the informal description in Fig. 8 where (i) the boxed terms are ignored, and (ii) the coefficients of all the secret matrices come from a distribution χ.

In this setting, the deniability property says that one can efficiently simulate the key $\mathbf{K}_\mathsf{B}$ and the ciphertext ct, given only the public key pk_B of B and the secret key sk_A of A. This means that no efficient adversary $\mathcal{A}(\mathsf{pk}_\mathsf{A}, \mathsf{pk}_\mathsf{B}, \mathsf{sk}_\mathsf{A})$, which has the public key as well as the secret key sk_A, can distinguish between the honestly computed $(\mathbf{K}_\mathsf{B}, \mathsf{ct})$ and the simulated one. By construction, if one can simulate $\mathbf{V}$, then ct and $\mathbf{K}_\mathsf{B}$ come for free, hence we focus on the latter task. Now, in order to prove deniability, we first observe that:

$$\begin{aligned}\mathbf{V} = \mathbf{S}_\mathsf{B}\mathbf{B}_\mathsf{A} + \mathbf{E}_\mathsf{B} &= \mathbf{S}_\mathsf{B}\mathbf{D}_\mathsf{A} + \mathbf{S}_\mathsf{B}\mathbf{A}\mathbf{S}_\mathsf{A} + \mathbf{E}_\mathsf{B} \\ &= \mathbf{S}_\mathsf{B}\mathbf{D}_\mathsf{A} - \mathbf{D}_\mathsf{B}\mathbf{S}_\mathsf{A} + \mathbf{B}_\mathsf{B}\mathbf{S}_\mathsf{A} + \mathbf{E}_\mathsf{B}.\end{aligned}$$

Since we are already given pk_B and sk_A, we only need to find a way to simulate the term $\mathbf{S}_\mathsf{B}\mathbf{D}_\mathsf{A} - \mathbf{D}_\mathsf{B}\mathbf{S}_\mathsf{A}$. A naive approach would be to simulate the secret key of B. That is, simply sample $\mathbf{S}_\mathsf{sim}, \mathbf{D}_\mathsf{sim} \leftarrow\$ \chi^{\bar{n}\times n}$ and set

$$\mathbf{V}_\mathsf{sim} := \mathbf{S}_\mathsf{sim}\mathbf{D}_\mathsf{A} - \mathbf{D}_\mathsf{sim}\mathbf{S}_\mathsf{A} + \mathbf{B}_\mathsf{B}\mathbf{S}_\mathsf{A} + \mathbf{E}_\mathsf{B}. \tag{1}$$

We formally show that this approach is a valid simulation under the LWE assumption. To this end, we define a (matrix-version) Extended-LWE (ELWE) problem which can be described as follows: distinguish between

$$(\mathbf{A}, \mathbf{S}\mathbf{A}+\mathbf{D}, \mathbf{Z}, \mathbf{W}, \mathbf{S}\mathbf{Z}+\mathbf{D}\mathbf{W}) \quad \text{and} \quad (\mathbf{A}, \mathbf{T}, \mathbf{Z}, \mathbf{W}, \mathbf{S}\mathbf{Z}+\mathbf{D}\mathbf{W}) \tag{2}$$

where $\mathbf{S}, \mathbf{D} \leftarrow\$ \chi^{n\times\bar{n}}, \mathbf{Z}, \mathbf{W} \leftarrow\$ \chi^{\bar{n}\times n}$ and $\mathbf{T}$ is uniformly random in $\mathbb{Z}_q^{\bar{n}\times n}$. We call $\mathbf{S}\mathbf{Z}+\mathbf{D}\mathbf{W}$ the hint matrix, and its height the number of hints. Now, the proof consists of two main steps which we sketch out below:

1. First, we argue that if we consider a hybrid Game_1, where we swap the public key $\mathbf{B}_\mathsf{B} := \mathbf{S}_\mathsf{B}\mathbf{A} + \mathbf{D}_\mathsf{B}$ with a uniformly random matrix over $\mathbb{Z}_q$, then no efficient adversary can distinguish between the two experiments under the ELWE assumption. Indeed, given the ELWE tuple $(\mathbf{A}, \mathbf{T}, \mathbf{Z}, \mathbf{H})$, the reduction $\mathcal{B}$ can set $\mathbf{T} := \mathbf{B}_\mathsf{B}$, $\mathbf{Z} := -\mathbf{D}_\mathsf{A}$, $\mathbf{W} := \mathbf{S}_\mathsf{A}$ and $\mathbf{V} := \mathbf{H} + \mathbf{B}_\mathsf{B}\mathbf{S}_\mathsf{A} + \mathbf{E}_\mathsf{B}$, where $\mathbf{E}_\mathsf{B}$ is sampled by $\mathcal{B}$ as in the protocol. Then, either $\mathbf{T} = \mathbf{S}_\mathsf{B}\mathbf{A} + \mathbf{D}_\mathsf{B}$ or $\mathbf{T}$ is uniformly random. However, in both cases we have $\mathbf{V} = \mathbf{H} + \mathbf{B}_\mathsf{B}\mathbf{S}_\mathsf{A} + \mathbf{E}_\mathsf{B} = \mathbf{S}_\mathsf{B}\mathbf{D}_\mathsf{A} - \mathbf{D}_\mathsf{B}\mathbf{S}_\mathsf{A} + \mathbf{B}_\mathsf{B}\mathbf{S}_\mathsf{A} + \mathbf{E}_\mathsf{B}$. Therefore, if $\mathbf{T}$ is a true LWE instance then the reduction can simulate the deniability game. Otherwise, $\mathbf{T}$ is uniformly random and thus $\mathcal{B}$ simulates Game_1. For notational convenience, in Game_1 we denote the secret key matrices for B as $(\mathbf{S}_\mathsf{sim}, \mathbf{D}_\mathsf{sim})$ rather than $(\mathbf{S}_\mathsf{B}, \mathbf{D}_\mathsf{B})$, which are *only* used to compute $\mathbf{V}$.

2. The next step is to introduce the next Game_2, where the public key is now computed as $\mathbf{B}_\mathsf{B} := \mathbf{S}_\mathsf{B}\mathbf{A} + \mathbf{D}_\mathsf{B}$, for $\mathbf{S}_\mathsf{B}, \mathbf{D}_\mathsf{B} \leftarrow\$ \chi^{\bar{n}\times n}$, instead of being sampled uniformly at random. Since $\mathbf{S}_\mathsf{B}, \mathbf{D}_\mathsf{B}$ are only used to compute $\mathbf{B}_\mathsf{B}$, computational indistinguishability of the last two games comes from the hardness of plain LWE. We highlight that Game_2 is not just reversing Game_1, since $\mathbf{V}$ is still computed as $\mathbf{V}_\mathsf{sim} = \mathbf{S}_\mathsf{sim}\mathbf{D}_\mathsf{A} - \mathbf{D}_\mathsf{sim}\mathbf{S}_\mathsf{A} + \mathbf{B}_\mathsf{B}\mathbf{S}_\mathsf{A} + \mathbf{E}_\mathsf{B}$.

Finally, one observes that the matrix $\mathbf{V}_\mathsf{sim}$ from (1) is computed exactly as in Game_2. Hence, the indistinguishability of $\mathbf{V}$ and $\mathbf{V}_\mathsf{sim}$ follows from both ELWE and LWE assumptions via the hybrid argument.

4.2 Achieving Stronger Security Properties

As discussed above, for the main applications in this paper we need our split-KEM to satisfy stronger security notion than e.g. IND-CPA [17]. It turns out that split-KEM deniability, which is a useful feature on its own, can be used as a building block to prove more advanced properties. We showcase this by giving an intuition on how to build a "decaps-OW-CPA" split-KEM.

In the decaps-OW-CPA game, the challenger generates the secret and public keys for both parties, applies the encapsulation for B to obtain $(\mathbf{K}^*, \mathsf{ct}^*) \leftarrow \mathsf{Encaps}(\mathsf{pk}_\mathsf{A}, \mathsf{sk}_\mathsf{B})$, and outputs $(\mathsf{pk}_\mathsf{A}, \mathsf{pk}_\mathsf{B}, \mathsf{ct}^*)$ to the adversary $\mathcal{A}$, which then returns a new ciphertext-key pair $(\mathsf{ct}', \mathbf{K}'_\mathsf{A})$. The adversary's goal is to guess the key $\mathbf{K}_\mathsf{A}$ which is output by $\mathsf{Decaps}(\mathsf{pk}_\mathsf{B}, \mathsf{sk}_\mathsf{A}, \mathsf{ct}')$. Note that by definition of deniability, we can use the simulator $\mathsf{Sim}(\mathsf{pk}_\mathsf{B}, \mathsf{sk}_\mathsf{A})$ from before to simulate the encapsulation output $(\mathbf{K}^*, \mathsf{ct}^*)$ instead. After the substitution, the only place, where the secret key sk_B of B is actually used, is when computing $\mathbf{B}_\mathsf{B} = \mathbf{S}_\mathsf{B}\mathbf{A} + \mathbf{D}_\mathsf{B}$. Hence, under the LWE assumption, we can swap $\mathbf{B}_\mathsf{B}$ with a uniformly random matrix. After all these changes, the modified decaps-OW-CPA experiment, which we call decaps-OW-CPA*, can be explicitly summarised below (ignoring the boxed items for now).

decaps-OW-CPA*

$\mathbf{S}_\mathsf{A}, \mathbf{D}_\mathsf{A} \leftarrow\!\$\ \chi^{n\times\bar{n}}; \mathbf{F}_\mathsf{A} \leftarrow\!\$\ \chi^{\bar{n}\times\bar{n}};\ \mathbf{B}_\mathsf{A} \leftarrow \mathbf{A}\mathbf{S}_\mathsf{A} + \mathbf{D}_\mathsf{A}$

$\mathbf{E}_\mathsf{B} \leftarrow\!\$\ \Psi^{\bar{n}\times\bar{n}};\ \mathbf{S}_\mathsf{sim}, \mathbf{D}_\mathsf{sim} \leftarrow\!\$\ \chi^{\bar{n}\times n}$

$\mathbf{V}_\mathsf{sim} \leftarrow \mathbf{S}_\mathsf{sim}\mathbf{D}_\mathsf{A} - \mathbf{D}_\mathsf{sim}\mathbf{S}_\mathsf{A} + \mathbf{B}_\mathsf{B}\mathbf{S}_\mathsf{A} + \mathbf{E}_\mathsf{B}$

$\mathsf{ct} \leftarrow \mathsf{HelpRec}(\mathbf{V}_\mathsf{sim})$

$\mathbf{K}'_\mathsf{A}, \mathsf{ct}' \leftarrow\!\$\ \mathcal{A}(\mathsf{pk}_\mathsf{A}, \mathsf{pk}_\mathsf{B}, \mathsf{ct})$

$\mathbf{V}' \leftarrow \mathbf{B}_\mathsf{B}\mathbf{S}_\mathsf{A} + \mathbf{F}_\mathsf{A}; \mathbf{K}_\mathsf{A} \leftarrow \mathsf{Rec}(\mathbf{V}', \mathsf{ct}')$

Our main goal is to make sure that after a certain number of hybrid games, $\mathbf{V}'$ is uniformly random. If this is the case, then using a high min-entropy argument on the reconciliation mechanism Rec, we could show that the probability of guessing $\mathbf{K}_\mathsf{A}$ is negligible.

By looking at how $\mathbf{V}' = \mathbf{B}_\mathsf{B}\mathbf{S}_\mathsf{A}$ is currently defined, the intuition is to somehow rely on the LWE problem. The first issue with this idea is that the expression for $\mathbf{V}'$ lacks an "error term". We can easily patch this by adding a short $\bar{n}\times\bar{n}$ matrix $\mathbf{F}_\mathsf{A} \leftarrow\!\$\ \chi^{\bar{n}\times\bar{n}}$ as a part of the secret of sk_A, which is used only for decapsulation. Then, by defining $\mathbf{V}'$ exactly as in the figure above, we have:

$$\begin{bmatrix}\mathbf{B}_\mathsf{A}\\ \mathbf{V}'\end{bmatrix} = \begin{bmatrix}\mathbf{A}\\ \mathbf{B}_\mathsf{B}\end{bmatrix}\mathbf{S}_\mathsf{A} + \begin{bmatrix}\mathbf{D}_\mathsf{A}\\ \mathbf{F}_\mathsf{A}\end{bmatrix}$$

which is exactly a LWE instance. The reason why we cannot swap $(\mathbf{B}_\mathsf{A}, \mathbf{V}')$ with random matrices is that we still need to simulate $\mathbf{V}_\mathsf{sim}$ which depends on the secret key sk_A. We solve this issue using the following approach. By rearranging, we first compute $\mathbf{V}'$ and then $\mathbf{V}$. Next, we can alternatively compute $\mathbf{V}$ as $\mathbf{V} = \mathbf{S}_\mathsf{sim}\mathbf{D}_\mathsf{A} - \mathbf{D}_\mathsf{sim}\mathbf{S}_\mathsf{A} + \mathbf{V}' + \mathbf{E}'$ where $\mathbf{E}' = \mathbf{E}_\mathsf{B} - \mathbf{F}_\mathsf{A}$. Here, our key idea is to apply *rejection sampling* [47]. Namely, if we allow to pick coefficients of $\mathbf{E}_\mathsf{B}$ uniformly at random from much larger interval $[-\beta, \beta]$ than for χ, i.e. $\psi := \mathcal{U}([-\beta, \beta]^{\bar{n}\times\bar{n}})$, then conditioned on $\|\mathbf{E}_\mathsf{B} + \mathbf{F}_\mathsf{A}\|_\infty \le \beta - \gamma$, where $\|\mathbf{F}_\mathsf{A}\|_\infty \le \gamma$, the matrix $\mathbf{E}' = \mathbf{E}_\mathsf{B} - \mathbf{F}_\mathsf{A}$ is uniformly random with coefficients between $[-\beta+\gamma, \beta-\gamma]$. This allows us to simply substitute the term $\mathbf{E}_\mathsf{B} - \mathbf{F}_\mathsf{A}$ with a uniformly random matrix $\mathbf{E}'$, which in particular is independent of $\mathbf{F}_\mathsf{A}$.

After all these modifications, we show that under a slightly modified Extended-LWE assumption, we can swap $(\mathbf{B}_\mathsf{A}, \mathbf{V}')$ with uniformly random matrices which finishes the proof. Here, the main change from (2) is that we now allow the error terms in the LWE sample not to be included in the hint. That is, this time the adversary has to distinguish between

$$\left(\begin{bmatrix}\mathbf{A}\\ \mathbf{B}\end{bmatrix}, \begin{bmatrix}\mathbf{A}\\ \mathbf{B}\end{bmatrix}\mathbf{S} + \begin{bmatrix}\mathbf{D}\\ \mathbf{F}\end{bmatrix}, \mathbf{Z}, \mathbf{W}, \mathbf{Z}\mathbf{S} + \mathbf{W}\mathbf{D}\right) \quad \text{and}$$
$$\left(\begin{bmatrix}\mathbf{A}\\ \mathbf{B}\end{bmatrix}, \mathbf{T}, \mathbf{Z}, \mathbf{W}, \mathbf{Z}\mathbf{S} + \mathbf{W}\mathbf{D}\right).$$

The modified scheme we use (with added noise in decapsulation) is given in Figure 8 with boxed instructions.

Hardness of Extended-LWE. We provide a formal reduction from the newly defined Extended-LWE problem to plain LWE. The reduction is similar to the one presented by Alperin-Sheriff and Peikert [2], albeit not a straightforward adaptation. The issue is that in [2], the ELWE problem outputs hints of size only a single $\mathbb{Z}_q$ element. Here, we have to deal with $\bar{n}$ many $\mathbb{Z}_q$-hints, which becomes problematic when arguing about invertibility of certain matrices when following the original reduction. We circumvent this by first reducing our ELWE to the corresponding knapsack-type version [52], then applying the reduction from [2] to reduce to plain knapsack-LWE, and finally going back to plain LWE using [52].

UNF-1KMA and IND-1BatchCCA. We have informally proven so far that the modified version of FrodoKEX given above is decaps-OW-CPA and OW-CPA. In the full version [25] we show that any scheme satisfying both these properties can easily be transformed into a UNF-1KMA and IND-1BatchCCA split-KEM in the ROM and QROM using the T_CH transform of Huguenin-Dumittan and Vaudenay [39] adapted to the split-KEM setting.

4.3 Concrete Instantiation

In Table 1 we propose the parameter set where we aim for 128-bit classical security. We detail in the full version [25] the calculations that lead to these numbers.

n	$\bar{n}$	q	B	χ	β	$\lvert pk \rvert$	$\lvert ct \rvert$
1452	8	31751	4	$\mathcal{U}(\{-1,1\})$	2^6	21.3KB	64B

Table 1: Concrete parameters for our lattice-based split-KEM. B is the number of bits that can be extracted per coefficient. We note that in practice, we do not need to include the whole matrix $\mathbf{A}$ in the public key, but rather the seed for the pseudo-random function to generate it.

5 Benchmarks, Comparison and Discussion

Hereafter, we refer to the scheme by Brendel et al. [16] as SPQR, and we refer to the deniable (i.e. with ring signatures) scheme by Hashimoto et al. [38] as HKKP.

5.1 Benchmarks

Security of non-standard primitive. As K-Waay, SPQR and HKKP can each be implemented using, except for a single primitive in each case, only (soon to be) standardised primitives, we wish to compare the security of the non-standard primitives. In the case of K-Waay it is a split-KEM, here implemented using a variant of FrodoKEX passed through the T_{CH} transform (that we call FrodoKEX+), and in the case of both HKKP and SPQR it is a ring signature (or a DVS derived from a ring signature). The authors of both SPQR and HKKP proposed possible implementations for the RS without picking one in particular. The most efficient one for a ring of size 2 that has an existing C implementation is Raptor [45] which we use for the benchmarks below. Other candidates would be Falafl [9] or DualRing-LB [64].

We present in Table 2 a summary of the security claims, approximate leading factor in the bounds in the (Q)ROM, and assumptions for these non-standard primitives. We note that none of these primitives are proven secure in the standard model and all are based on lattices.

First, we note that parameters for these RS schemes are picked *before* the reduction in the ROM. That is, a primitive P based on lattices is built, parameters are chosen such that P satisfies the security claim, then P is used to build a RS in the ROM, which incurs a loss factor that usually depends on the number of queries to the random oracle q_H. In particular, it is common to have at least a q_H factor in the security bound (e.g. if the adversary can make 2^{32} queries to the RO, the security level is reduced by 32 bits). Therefore, the claimed security level does not match the *provable* security level. In the QROM, the security loss is usually greater: square root and q_H^2 or q_H^3 losses are quite common, however these schemes are not proven secure in this model.

We chose the opposite approach in designing a split-KEM with a conservative assumption (i.e., plain LWE) and parameters. Therefore, FrodoKEX+ with our proposed parameters achieve 128 (resp. 64) bits of classical (resp. quantum) security *after* the (Q)ROM proof. We put the (approximate) highest terms of both the ROM and QROM security bounds in Table 2. These satisfy our security claims as long as $q_H \leq 2^{64}$ in the ROM and $q_H \cdot d \leq 2^{64}$ in the QROM, where d is the number of public key/ciphertext tuples allowed in the IND-1BatchCCA game. We note that in K-Waay, d corresponds to the number of distinct users trying to communicate with an offline receiver after all prekeys have run out, thus should typically be small. We note that the leading coefficients in all our bounds are always related to the (Q)ROM security (e.g. finding a collision) and not to the underlying hardness assumptions. Therefore, one can easily make these coefficients smaller by picking a larger digest size n, e.g. 512 bits.

The reason behind the approximations and lack of QROM proofs for PQ ring signatures is likely the youth of the field and the speed at which it is evolving. Still, we believe it is worth noting as it makes any comparison between our protocol and previous ones quite difficult.

Benchmarking. The protocols we benchmarked are: our own implementation of the current X3DH protocol, a witness protocol made only with PQ KEMs and a signature scheme similar to the *non-deniable* variant of HKKP, Brendel et al.'s [16] construction SPQR using PQ KEMs, a signature scheme and a DVS, Hashimoto et al.'s [37, 38] construction HKKP using PQ KEMs, a signature scheme and a RS, and our scheme K-Waay with FrodoKEX+ using PQ split-KEM, KEMs and a signature scheme.

We picked Kyber512 as the KEM, both Falcon-512 and Dilithium2 as signatures, and Raptor as the ring signature. We implemented both HKKP and SPQR with signed prekeys as is the case in Signal's implementation of X3DH. That is, a PQ signature key pair is part of the long-term key, and ephemeral keys uploaded to server are signed with it. Note that this is make explicit in K-Waay as the ephemeral keys are signed with the long-term one. The authors of HKKP show that this is not necessary in their protocol, however not doing so weakens perfect forward secrecy.

We built the different protocols in C using the `liboqs` library[4] for Kyber, Falcon, and Dilithium, the Raptor implementation provided by the authors[5], and a modified version of the `lwe-frodo` library[6] with scaled parameters to properly simulate FrodoKEX+. More precisely, the modulus was set to the first power of 2 larger than the modulus in FrodoKEX+, the addition of the noise during decapsulation was also added, and the noise distributions were modified to match the ones of FrodoKEX+. We did not optimise the scheme in any way (e.g. by using AVX instructions or parallelisation) and we leave this as future work. For the sake of completeness, we also provide a reference implementation of FrodoKEX+ in Rust[7] for the interested reader. All benchmarks were run on a

[4] https://github.com/open-quantum-safe/liboqs
[5] https://github.com/zhenfeizhang/raptor
[6] https://github.com/lwe-frodo/lwe-frodo
[7] https://github.com/lehugueni/frodokexp-rust

Scheme	Cl. (C)	Cl. (Q)	ROM bnd	QROM bnd	Assumption
FrodoKEX+	128	64	$q_H^2/2^{256}$	$q_H \cdot d/2^{128}$	LWE
Raptor [45]	114	103	?	✗	NTRU
DualRing-LB [64]	(128)	(64)	?	✗	MSIS, MLWE
Falafl [9]	128	64	?	✗	MSIS, MLWE

Table 2: Security comparison between FrodoKEX+ and several post-quantum RS. 'Cl.' stands for claimed number of security bits. DualRing-LB's authors do not seem to make a clear security claim, we thus assume NIST level I. '?' indicates that no bound is explicitly given for the security, '✗' indicates that no proof is provided in the QROM.

virtual machine running Ubuntu 22.04 with 2 cores of an Intel i7-9750H running at 2.60GHz and 4GB of RAM allocated.

Speed. For the speed benchmark, we measured how many cycles each protocol takes in one execution. We summarise our results in a logarithmic graph on Figure 9 (note that the internal division of the bars is linear).

Fixing the choice of KEM and signature scheme, our protocol K-Waay, depending on the choice of KEM and signature scheme, is between 3 and 6 times faster than the previous proposals even with our relatively conservative parameter choice. In our protocol K-Waay using Dilithium2, most cycles are spent in the ephemeral key generation, while using Falcon makes the static key generation as expensive as the ephemeral key one. Overall, one can see that Falcon, while more compact than Dilithium2, has a great impact on efficiency. For instance, K-Waay with Dilithium2 is faster than the non-deniable scheme using Kyber and Falcon.

Apart from Falcon, we see that the most time-consuming primitives are the non-standard ones, i.e., ring signatures and split-KEM. Hence, we see that the KEM+SIG protocol (HKKP's baseline proposal) performs even better than X3DH, which shows once again that lattice-based schemes can be faster than their classical counterparts. However, we recall that it does not provide deniability. At last, we note that X3DH is the only construction that spends more time in sending and receiving than generating keys. Our protocol's Send and Receive (i.e. BatchReceive with a single input message) procedures are very fast.

Data size. In Table 3, we provide for each scheme the size of the long-term keys, the prekeys (output by Init in our DAKE syntax), and the ciphertext output by the sender. We computed for both HKKP and SPQR the size with and without long-term signatures. We see that K-Waay compares well in term of long-term public key and ciphertext size as both are smaller than signed HKKP and SPQR. However, the prekeys are much larger as one could expect from a LWE-based scheme and due to our conservative choice of parameters.

Scheme	|lpk|	|prek|	|ct|
K-Waay + Dilithium	2112	24520	1632
K-Waay + Falcon	1697	22790	1632
HKKP [38]	1700	1700	4056
HKKP [38] + Dilithium2	3012	4120	4056
HKKP [38] + Falcon	2597	2390	4056
SPQR [16]	3400	1632	4824
SPQR [16] + Dilithium2	4712	4052	4824
SPQR [16] + Falcon	4297	2322	4824

Table 3: Size comparison in bytes between K-Waay instantiated with FrodoKEX+, HKKP [38] and SPQR [16]. We also computed the sizes for both HKKP and SPQR implemented with signed prekey bundles.

5.2 Advantages, Limitations and Discussion

Running out of ephemeral keys. The main disadvantage of our protocol is that running out of ephemeral keys requires the receiver to abort if *any* of the sessions that used the same prekey is bogus. If this happens, then a malicious party could mount some kind of denial of service (DoS) attack against the user that was offline for too long by sending a bogus split-KEM ciphertext. There is an obvious trade-off between the risk of such an attack happening and the number of ephemeral keys uploaded on the server, thus the storage required on the server. We leave the analysis and the mitigation of such a threat as future work, but we believe that if a reasonable amount of prekeys are uploaded, creating fake accounts is difficult (e.g. by requiring a phone number as in Signal), and/or users are online often enough, such an attack would be difficult to mount. Furthermore, several practical mitigations are possible. For instance, if the receiver (i.e. the victim) received a bogus ciphertext among the n ciphertexts sent for the same prekey while offline, they can restart K-Waay with the n parties but as the initiator, which will probably succeed. The victim could also send n new prekeys to the n initiators directly, making sure the protocol will succeed at the next iteration. This would make the attack less useful as it could only *delay* communication and not prevent it.

We also think it is worth mentioning that the trick we propose might be easy to misimplement. In particular, it is crucial that no information about which split-KEM ciphertext

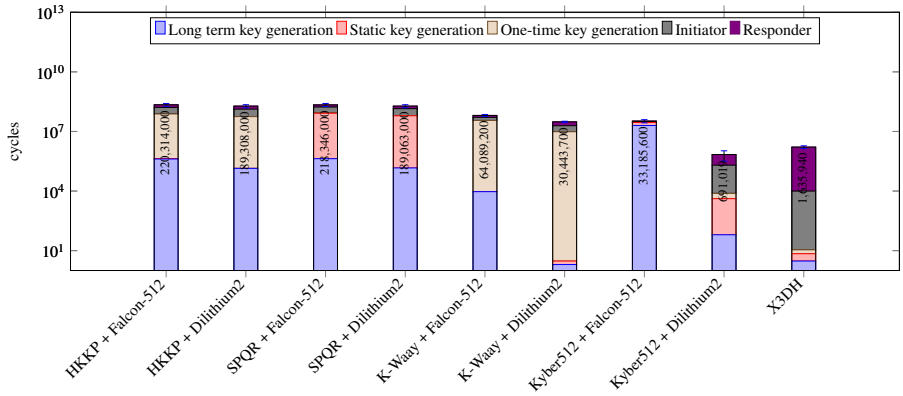

Figure 9: Speed benchmark for X3DH protocols

failed leaks if such a situation occurs. That is, all precautions should be taken such that such leakage via side-channels is prevented.

split-KEM instead of Ring Signatures. The first advantage of using our protocol over existing ones is the use of an ephemeral primitive instead of a long-term one, the former being often more efficient as the security requirements are less strict. In addition, the use of a primitive similar to a post-quantum KEM allows us to leverage the extensive literature on the topic and existing safe/optimised implementations. This also gives good security guarantees as post-quantum KEMs have been heavily scrutinised as part of the NIST standardisation process. For example, as mentioned above, our proposed lattice-based implementation is based on a key-exchange variant of FrodoKEM, which is itself the PQ KEM recommended by the German Federal Office for Information Security (BSI) [35]. Overall we think that a split-KEM such as FrodoKEX+ is more mature and closer to being usable in practice than ring signatures.

On the necessity of modifying FrodoKEX. Currently, our split-KEM significantly differs from the original FrodoKEX in two aspects: (i) the modulus for our construction has to be prime in order for our reduction from Extended-LWE to LWE to hold[8], and (ii) we have to introduce additional masking terms to prove UNF-1KMA security. However, we believe that both changes are artefacts of the security proofs, and the original FrodoKEX split-KEM should be (up to a reasonable security loss) deniable.

There are alternative reduction techniques from Extended-LWE to LWE in the literature [14, 15], which do not rely on having an odd modulus at the cost of using discrete Gaussian error distributions with large parameters. It is thus an interesting research problem to efficiently reduce Extended-LWE to LWE for even modulus with small reduction loss. In practice, the most efficient LWE attacks do not consider the structure of the modulus, so intuitively this should translate to the Extended-LWE setting[9].

As for our second main modification, it is unclear how to argue UNF-1KMA security without the additional masking terms and rejection sampling, even though the latter could potentially be removed using very recent techniques [1, 41]. Hence, we leave deniability and UNF-1KMA security of the original FrodoKEX construction as exciting future work.

Deniability. While the signature on the ephemeral public keys might give the impression that our protocol is less deniable than X3DH or previous PQ alternatives, this is actually not the case. The reason is that prekey bundles in these protocols are signed as well, but this detail is abstracted away in the analysis (i.e. it is assumed that all parties have received and authenticated all public-keys before the protocol actually starts). While this kind of analysis allows for strong deniability claims, in practice these protocols do not satisfy something stronger than some kind of peer-deniability. The exception is the ring signature based variant by Hashimoto et al. [38], where the prekey bundle is not necessarily signed. However, in this variant, the authors can only prove the security of their protocol in a weaker model (i.e. it satisfies a weaker notion of forward secrecy). Overall, if deniability should not come at the price of security, peer-deniability seems like the best notion one can achieve in these DAKEs.

We wished to provide a transparent model for peer-deniability, where the upload of signed ephemeral keys is made explicit. We also strengthen the deniability definition of Brendel et al. [16] by allowing the exposure of one of the parties (i.e. the receiving one, which would be the malicious party trying to frame the sender). While our protocol satisfies our stronger (in term of key exposure) notion of deniability, we believe both previous PQ X3DH alternatives satisfy it as

[8] Recall FrodoKEX [12] uses a power-of-two modulus for efficiency.

[9] Recently, various frameworks have been developed [28, 29], which measure concrete hardness of LWE given hints of specific form, such as linear combination of secrets with short random coefficients.

well. Indeed, in these schemes, the ephemeral keys are KEM and RS keys only, which are deniable. Hence, exposing these should not harm deniability.

Hashimoto et al. [38] consider a strong notion of deniability where the adversary is malicious (i.e. can arbitrarily deviate from the protocol) and show how to modify HKKP s.t. it is secure against such a threat. However, such deniability comes at the expense of NIZKs, which are complex, expensive and are not always proven secure in the QROM when random oracles are used. Moreover, as in other deniable systems against malicious adversaries, non-falsifiable assumptions (i.e. knowledge-type assumptions) are required to prove the security. In addition, it seems difficult to defend against adversaries actively trying to frame a given user in messaging in practice [24, 36]; for example, an adversary could also simply ask questions that would identify the victim with good probability. Because of these reasons, we do not consider such a notion of deniability here.

To contextualise our results, we remark here that cryptographic deniability, which is targeted by this work and all previous work on deniable X3DH key exchange, translates to deniability on a *system* level only if the application preserves deniability. For example, Collins et al. [24] observe that Signal as currently deployed does not provide this kind of 'practical' deniability for ordinary users. Suppose Bob is trying to frame Alice and hands over their phone, that contains a transcript of communication between Alice and Bob, to a judge. Because Signal authenticates users (either directly or indirectly through Signal sealed sender [46]), unless Bob was able to modify their phone (which depends on the technical expertise of Bob), the judge can deduce that the conversation plausibly took place as in the transcript, regardless of the cryptographic protocols employed underneath. It is interesting future work to further explore deniability on the broader system level.

An optimisation. As presented in Section 3, the K-Waay protocol generates a signature for each ephemeral public-key uploaded. This can easily be optimised by signing the whole prekey bundle containing several ephemeral keys. This way, the server needs to store only one PQ signature for each user. The downside is that now each user needs to download the whole bundle to verify the signature. This offers a trade-off between data stored at the server and sent to clients.

Strengthening protocol security. Note that we proved K-Waay key indistinguishable in a model that considers state exposures like Hashimoto et al.'s [38] but is nonetheless weaker (our protocol is provably-secure under a notion similar to that of Brendel et al. [16]). This is mainly since our protocol only uses ephemeral split-KEM keys. As noted by Brendel et al. [17], however, it seems much more difficult to construct split-KEM secure under several encapsulation/decapsulation queries, which we leave again as important future work.

Future work. An interesting line of research would be to try to build other unforgeable IND-1BatchCCA split-KEMs that are more efficient (mostly in key and ciphertext size). One obvious direction would be to work over structured lattices [43, 49, 59]. Indeed, Ring/Module-LWE with hints (similar to our Extended-LWE problem) have already been analysed from a theoretical point of view [14, 51]. We also believe that our techniques can also be applied in the ring setting. However, for security purposes one needs to take a ring dimension d to be at least linear in the security parameter λ which becomes problematic when proving deniability. Indeed, the leaked hint is informally the product of secret keys of both parties. Thus, in the ring setting the hint would be at least a single polynomial, which contains $d = O(\lambda)$ coefficients. We predict that this would result with much larger reduction loss than what we have now. However, the concrete analysis is left as future work.

Finally, on a more practical side, it would be informative to benchmark our protocol and others in a real-life scenario or something close to it, and to implement other ring signatures schemes to have a more complete comparison. Different parameter sets to achieve higher level of security could also be provided and benchmarked.

Acknowledgements

Loïs Huguenin-Dumittan is supported by a grant (project no. 192364) of the Swiss National Science Foundation (SNSF).

References

[1] Shweta Agrawal, Damien Stehlé, and Anshu Yadav. Round-optimal lattice-based threshold signatures, revisited. In *ICALP*, volume 229 of *LIPIcs*, pages 8:1–8:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022.

[2] Jacob Alperin-Sheriff and Chris Peikert. Circular and KDM security for identity-based encryption. In *Public Key Cryptography*, volume 7293 of *Lecture Notes in Computer Science*, pages 334–352. Springer, 2012.

[3] Joël Alwen, Bruno Blanchet, Eduard Hauck, Eike Kiltz, Benjamin Lipp, and Doreen Riepel. Analysing the hpke standard. In *EUROCRYPT 2021*, pages 87–116. Springer, 2021.

[4] Joël Alwen, Sandro Coretti, and Yevgeniy Dodis. The double ratchet: Security notions, proofs, and modularization for the signal protocol. In *EUROCRYPT*, pages 129–158. Springer, 2019.

[5] Christoph Bader, Dennis Hofheinz, Tibor Jager, Eike Kiltz, and Yong Li. Tightly-secure authenticated key exchange. In Yevgeniy Dodis and Jesper Buus

Nielsen, editors, *Theory of Cryptography - 12th Theory of Cryptography Conference, TCC 2015, Warsaw, Poland, March 23-25, 2015, Proceedings, Part I*, volume 9014 of *Lecture Notes in Computer Science*, pages 629–658. Springer, 2015.

[6] Mihir Bellare. New proofs for nmac and hmac: Security without collision-resistance. In *CRYPTO 2006*, pages 602–619. Springer, 2006.

[7] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *CCS*, pages 62–73. ACM, 1993.

[8] Daniel J Bernstein. Curve25519: new diffie-hellman speed records. In *International Workshop on Public Key Cryptography*, pages 207–228. Springer, 2006.

[9] Ward Beullens, Shuichi Katsumata, and Federico Pintore. Calamari and falafl: Logarithmic (linkable) ring signatures from isogenies and lattices. In *ASIACRYPT (2)*, volume 12492 of *Lecture Notes in Computer Science*, pages 464–492. Springer, 2020.

[10] Alexander Bienstock, Jaiden Fairoze, Sanjam Garg, Pratyay Mukherjee, and Srinivasan Raghuraman. A more complete analysis of the signal double ratchet algorithm. In Yevgeniy Dodis and Thomas Shrimpton, editors, *CRYPTO 2022*, volume 13507 of *Lecture Notes in Computer Science*, pages 784–813. Springer, 2022.

[11] Dan Boneh, Özgür Dagdelen, Marc Fischlin, Anja Lehmann, Christian Schaffner, and Mark Zhandry. Random oracles in a quantum world. In *ASIACRYPT*, volume 7073 of *Lecture Notes in Computer Science*, pages 41–69. Springer, 2011.

[12] Joppe W. Bos, Craig Costello, Léo Ducas, Ilya Mironov, Michael Naehrig, Valeria Nikolaenko, Ananth Raghunathan, and Douglas Stebila. Frodo: Take off the ring! practical, quantum-secure key exchange from LWE. In *CCS*, pages 1006–1018. ACM, 2016.

[13] Joppe W. Bos, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, John M. Schanck, Peter Schwabe, Gregor Seiler, and Damien Stehlé. CRYSTALS - kyber: A cca-secure module-lattice-based KEM. In *2018 IEEE European Symposium on Security and Privacy, EuroS&P*, pages 353–367, 2018.

[14] Katharina Boudgoust, Corentin Jeudy, Adeline Roux-Langlois, and Weiqiang Wen. On the hardness of module-lwe with binary secret. In *CT-RSA*, volume 12704 of *Lecture Notes in Computer Science*, pages 503–526. Springer, 2021.

[15] Zvika Brakerski, Adeline Langlois, Chris Peikert, Oded Regev, and Damien Stehlé. Classical hardness of learning with errors. *CoRR*, abs/1306.0281, 2013.

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

[17] Jacqueline Brendel, Marc Fischlin, Felix Günther, Christian Janson, and Douglas Stebila. Towards post-quantum security for signal's X3DH handshake. In *SAC*, volume 12804 of *Lecture Notes in Computer Science*, pages 404–430. Springer, 2020.

[18] Ran Canetti, Palak Jain, Marika Swanberg, and Mayank Varia. Universally composable end-to-end secure messaging. In Yevgeniy Dodis and Thomas Shrimpton, editors, *CRYPTO 2022*, volume 13508 of *Lecture Notes in Computer Science*, pages 3–33. Springer, 2022.

[19] Ran Canetti and Hugo Krawczyk. Security analysis of ike's signature-based key-exchange protocol. In *CRYPTO 2002*, pages 143–161. Springer, 2002.

[20] Wouter Castryck and Thomas Decru. An efficient key recovery attack on sidh. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 423–447. Springer, 2023.

[21] Wouter Castryck, Tanja Lange, Chloe Martindale, Lorenz Panny, and Joost Renes. CSIDH: an efficient post-quantum commutative group action. In Thomas Peyrin and Steven D. Galbraith, editors, *ASIACRYPT 2018*, volume 11274 of *Lecture Notes in Computer Science*, pages 395–427. Springer, 2018.

[22] Katriel Cohn-Gordon, Cas Cremers, Benjamin Dowling, Luke Garratt, and Douglas Stebila. A formal security analysis of the signal messaging protocol. *J. Cryptol.*, 33(4):1914–1983, 2020.

[23] Katriel Cohn-Gordon, Cas Cremers, Kristian Gjøsteen, Håkon Jacobsen, and Tibor Jager. Highly efficient key exchange protocols with optimal tightness. In *CRYPTO 2019*, pages 767–797. Springer, 2019.

[24] Daniel Collins, Simone Colombo, and Loïs Huguenin-Dumittan. Real world deniability in messaging. *Cryptology ePrint Archive*, 2023.

[25] Daniel Collins, Loïs Huguenin-Dumittan, Ngoc Khanh Nguyen, Nicolas Rolin, and Serge Vaudenay. K-waay: Fast and deniable post-quantum x3dh without ring signatures. Cryptology ePrint Archive, 2023.

[26] Cas Cremers and Michele Feltz. One-round strongly secure key exchange with perfect forward secrecy and deniability. Cryptology ePrint Archive, Paper 2011/300, 2011.

[27] Cas Cremers and Mang Zhao. Secure messaging with strong compromise resilience, temporal privacy, and immediate decryption. In *IEEE S&P*, 2024.

[28] Dana Dachman-Soled, Léo Ducas, Huijing Gong, and Mélissa Rossi. LWE with side information: Attacks and concrete security estimation. In *CRYPTO (2)*, volume 12171 of *Lecture Notes in Computer Science*, pages 329–358. Springer, 2020.

[29] Dana Dachman-Soled, Huijing Gong, Tom Hanson, and Hunter Kippen. Revisiting security estimation for lwe with hints from a geometric perspective. In Helena Handschuh and Anna Lysyanskaya, editors, *CRYPTO 2023*, pages 748–781, Cham, 2023. Springer Nature Switzerland.

[30] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Trans. Inf. Theory*, 22(6):644–654, 1976.

[31] Samuel Dobson and Steven D. Galbraith. Post-quantum signal key agreement from SIDH. In Jung Hee Cheon and Thomas Johansson, editors, *Post-Quantum Cryptography - 13th International Workshop, PQCrypto 2022, Virtual Event, September 28-30, 2022, Proceedings*, volume 13512 of *Lecture Notes in Computer Science*, pages 422–450. Springer, 2022.

[32] Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, Peter Schwabe, Gregor Seiler, and Damien Stehlé. Crystals-dilithium: A lattice-based digital signature scheme. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2018(1):238–268, 2018.

[33] Cynthia Dwork, Moni Naor, and Amit Sahai. Concurrent zero-knowledge. *Journal of the ACM (JACM)*, 51(6):851–898, 2004.

[34] Muhammed F. Esgin, Ron Steinfeld, and Raymond K. Zhao. Matrict$^+$: More efficient post-quantum private blockchain payments. In *IEEE Symposium on Security and Privacy*, pages 1281–1298. IEEE, 2022.

[35] BSI German Federal Office for Information Security. Bsi tr-01102-1, 2023. https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/Publications/TechGuidelines/TG02102/BSI-TR-02102-1.html.

[36] Lachlan J Gunn, Ricardo Vieitez Parra, and N Asokan. Circumventing cryptographic deniability with remote attestation. *Proceedings on Privacy Enhancing Technologies*, 3:350–369, 2019.

[37] Keitaro Hashimoto, Shuichi Katsumata, Kris Kwiatkowski, and Thomas Prest. An efficient and generic construction for signal's handshake (X3DH): post-quantum, state leakage secure, and deniable. In Juan A. Garay, editor, *Public-Key Cryptography - PKC 2021 - 24th IACR International Conference on Practice and Theory of Public Key Cryptography, Virtual Event, May 10-13, 2021, Proceedings, Part II*, volume 12711 of *Lecture Notes in Computer Science*, pages 410–440. Springer, 2021.

[38] Keitaro Hashimoto, Shuichi Katsumata, Kris Kwiatkowski, and Thomas Prest. An efficient and generic construction for signal's handshake (X3DH): post-quantum, state leakage secure, and deniable. *J. Cryptol.*, 35(3):17, 2022.

[39] Loïs Huguenin-Dumittan and Serge Vaudenay. On ind-qcca security in the rom and its applications: Cpa security is sufficient for tls 1.3. In *EUROCRYPT 2022*, pages 613–642. Springer, 2022.

[40] Eike Kiltz, Jiaxin Pan, Doreen Riepel, and Magnus Ringerud. Multi-user cdh problems and the concrete security of naxos and hmqv. In *Cryptographers' Track at the RSA Conference*, pages 645–671. Springer, 2023.

[41] Duhyeong Kim, Dongwon Lee, Jinyeong Seo, and Yongsoo Song. Toward practical lattice-based proof of knowledge from hint-mlwe. In *CRYPTO (5)*, volume 14085 of *Lecture Notes in Computer Science*, pages 549–580. Springer, 2023.

[42] Brian LaMacchia, Kristin Lauter, and Anton Mityagin. Stronger security of authenticated key exchange. In *Provable Security: First International Conference, ProvSec 2007, Wollongong, Australia, November 1-2, 2007. Proceedings 1*, pages 1–16. Springer, 2007.

[43] Adeline Langlois and Damien Stehlé. Worst-case to average-case reductions for module lattices. *Des. Codes Cryptogr.*, 75(3):565–599, 2015.

[44] Xingye Lu, Man Ho Au, and Zhenfei Zhang. Raptor: A practical lattice-based (linkable) ring signature. In *ACNS*, volume 11464 of *Lecture Notes in Computer Science*, pages 110–130. Springer, 2019.

[45] Xingye Lu, Man Ho Au, and Zhenfei Zhang. Raptor: a practical lattice-based (linkable) ring signature. In *Applied Cryptography and Network Security: 17th International Conference, ACNS 2019, Bogota, Colombia, June 5–7, 2019, Proceedings 17*, pages 110–130. Springer, 2019.

[46] Joshua Lund. Technology preview: Sealed sender for signal. https://signal.org/blog/sealed-sender/, 2018. Last visited on 13-09-2023.

[47] Vadim Lyubashevsky. Fiat-shamir with aborts: Applications to lattice and factoring-based signatures. In *ASIACRYPT*, volume 5912 of *Lecture Notes in Computer Science*, pages 598–616. Springer, 2009.

[48] Vadim Lyubashevsky and Ngoc Khanh Nguyen. BLOOM: bimodal lattice one-out-of-many proofs and applications. In *ASIACRYPT (4)*, volume 13794 of *Lecture Notes in Computer Science*, pages 95–125. Springer, 2022.

[49] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. In *EUROCRYPT*, volume 6110 of *Lecture Notes in Computer Science*, pages 1–23. Springer, 2010.

[50] Moxie Marlinspike and Trevor Perrin. The x3dh key agreement protocol. *Open Whisper Systems*, 283, 2016.

[51] Jose Maria Bermudo Mera, Angshuman Karmakar, Tilen Marc, and Azam Soleimanian. Efficient lattice-based inner-product functional encryption. In *Public Key Cryptography (2)*, volume 13178 of *Lecture Notes in Computer Science*, pages 163–193. Springer, 2022.

[52] Daniele Micciancio and Petros Mol. Pseudorandom knapsacks and the sample complexity of LWE search-to-decision reductions. In *CRYPTO*, volume 6841 of *Lecture Notes in Computer Science*, pages 465–484. Springer, 2011.

[53] Chris Peikert. Lattice cryptography for the internet. In *PQCrypto*, volume 8772 of *Lecture Notes in Computer Science*, pages 197–219. Springer, 2014.

[54] Trevor Perrin and Moxie Marlinspike. The double ratchet algorithm. *GitHub wiki*, 2016.

[55] Mario Di Raimondo, Rosario Gennaro, and Hugo Krawczyk. Deniable authentication and key exchange. In Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati, editors, *Proceedings of the 13th ACM Conference on Computer and Communications Security, CCS 2006, Alexandria, VA, USA, October 30 - November 3, 2006*, pages 400–409. ACM, 2006.

[56] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In *STOC*, pages 84–93. ACM, 2005.

[57] Peter Schwabe, Douglas Stebila, and Thom Wiggers. Post-quantum TLS without handshake signatures. In Jay Ligatti, Xinming Ou, Jonathan Katz, and Giovanni Vigna, editors, *CCS '20: 2020 ACM SIGSAC Conference on Computer and Communications Security, Virtual Event, USA, November 9-13, 2020*, pages 1461–1480. ACM, 2020.

[58] Peter W. Shor. Polynominal time algorithms for discrete logarithms and factoring on a quantum computer. In Leonard M. Adleman and Ming-Deh A. Huang, editors, *Algorithmic Number Theory, First International Symposium, ANTS-I, Ithaca, NY, USA, May 6-9, 1994, Proceedings*, volume 877 of *Lecture Notes in Computer Science*, page 289. Springer, 1994.

[59] Damien Stehlé, Ron Steinfeld, Keisuke Tanaka, and Keita Xagawa. Efficient public key encryption based on ideal lattices. In *ASIACRYPT*, volume 5912 of *Lecture Notes in Computer Science*, pages 617–635. Springer, 2009.

[60] Nik Unger and Ian Goldberg. Deniable key exchanges for secure messaging. In Indrajit Ray, Ninghui Li, and Christopher Kruegel, editors, *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, Denver, CO, USA, October 12-16, 2015*, pages 1211–1223. ACM, 2015.

[61] Nik Unger and Ian Goldberg. Improved strongly deniable authenticated key exchanges for secure messaging. *Proc. Priv. Enhancing Technol.*, 2018(1):21–66, 2018.

[62] US National Security Agency. Announcing the commercial national security algorithm suite 2.0. https://media.defense.gov/2022/Sep/07/2003071834/-1/-1/0/CSA_CNSA_2.0_ALGORITHMS_.PDF.

[63] Nihal Vatandas, Rosario Gennaro, Bertrand Ithurburn, and Hugo Krawczyk. On the cryptographic deniability of the signal protocol. In Mauro Conti, Jianying Zhou, Emiliano Casalicchio, and Angelo Spognardi, editors, *ACNS 2020*, volume 12147 of *Lecture Notes in Computer Science*, pages 188–209. Springer, 2020.

[64] Tsz Hon Yuen, Muhammed F Esgin, Joseph K Liu, Man Ho Au, and Zhimin Ding. Dualring: generic construction of ring signatures with efficient instantiations. In *CRYPTO 2021*, pages 251–281. Springer, 2021.