Keeping Up with the KEMs: Stronger Security Notions for KEMs

Version 0.1.1, December 22, 2023*

Cas Cremers, Alexander Dax, and Niklas Medinger

CISPA Helmholtz Center for Information Security
{cremers,alexander.dax,niklas.medinger}@cispa.de

Abstract

Key Encapsulation Mechanisms (KEMs) are a critical building block for hybrid encryption and modern security protocols, notably in the post-quantum setting. Given the asymmetric public key of a recipient, the primitive establishes a shared secret key between sender and recipient. In recent years, a large number of abstract designs and concrete implementations of KEMs have been proposed, notably in the context of the NIST selection process for post-quantum primitives. The traditional security notion for KEMs has been the IND-CCA notion that was designed for public-key encryption (PKE). In recent work [17] additional properties, such as robustness and anonymity, were lifted from the PKE setting to the KEMs setting.

In this work we introduce several stronger security notions for KEMs. Our new properties formalize in which sense outputs of the KEM uniquely determine, i.e., *bind*, other values. Our new notions are based on two orthogonal observations: First, unlike PKEs, KEMs establish a unique key, which leads to natural binding properties for the established keys. Our new binding properties can be used, e.g., to prove the absence of attacks that were not captured by prior security notions, such as re-encapsulation attacks. If we regard KEMs as one-pass key exchanges, our key-binding properties correspond to implicit key agreement properties. Second, to prove the absence of weak keys, we have to consider not only honestly generated key pairs but also adversarially-generated key pairs.

We define a hierarchy of security notions for KEMs based on our observations. We position properties from the literature [17] within our hierarchy, provide separating examples, and give examples of real world KEMs in the context of our hierarchy.

1 Introduction

A Key Encapsulation Mechanism (KEM) [10] is a common building block in security protocols and cryptographic primitives such as hybrid encryption. Intuitively, a KEM can be seen as a specialized version of Public Key Encryption (PKE) that, instead of encrypting a payload message, specifically serves to generate and share a symmetric key between sender and recipient. A naive instantiation of a KEM could simply generate a symmetric key and encrypt this with a PKE. Whereas PKE schemes need to deal with arbitrary-length messages that might repeat, KEMs can be more optimized for their specific purpose.

During the last decade, many post-quantum secure KEMs have been proposed, see e.g., [2–5, 7, 8, 14, 19, 22, 24]. As a result, the popularity of KEMs as a building block in post-quantum security protocols has surged, where they are commonly used to replace Diffie-Hellman constructions for which no practical post-quantum secure scheme is currently available.

The traditional security notion for a KEM is a version of IND-CCA that is directly inherited from its related PKE notion. This essentially formalizes that given a KEM ciphertext, an adversary that does not have the corresponding private key cannot tell the difference between a random key and the encapsulated key. This is considered the only core requirement on new KEM designs.

Additionally, robustness-like properties have been proposed for KEMs in [17], which similarly inherit from their PKE counterparts. Initially, "robustness" [1] was defined in the PKE setting as the difficulty of finding a ciphertext valid under two different encryption keys. Phrased differently, a PKE is robust

*We provide a list of main changes in Appendix B.

if a ciphertext "binds to" (only decrypts under) one key. Because of the differences between so-called implicitly and explicitly rejecting PKEs/KEMs, such a strong notion might be unachievable for some designs, and hence [17] introduces a stronger SROB (strong robustness) notion, and a weaker SCFR (strong collision freeness) notion that can be met by a larger class of KEM designs.

In this work, we more systematically explore the possible binding properties of KEMs. Our work is similar in spirit to explorations in the space of digital signatures [9, 12, 20, 25] and authenticated encryption [11, 13, 16, 21], where recent works have identified many desirable binding properties for these primitives that could have prevented real-world attacks.

Our systematic analysis leads to the formulation of five core binding properties for KEMs, each of which comes in two variants. We then show how existing notions such as SROB and SCFR fit into our generic scheme. Whereas traditional KEM binding properties focused on binding values to a specific ciphertext, we propose variants that bind values to a specific key. We argue this is much more in line with viewing a KEM as a one-pass key exchange. Similarly, implicitly rejecting KEMs resemble implicitly authenticated key exchanges, where correct binding properties of the established key prevent classes of unknown key share attacks [6]. We relate our properties to properties previously reported in the literature, as well as related notions such as contributory behavior. We provide a full hierarchy for our properties with implications and separating examples.

Notably, we show an attack on an example key exchange protocol in the Kyber documentation when instantiated with another KEM, which proves that the protocol design in fact relies on properties of the used KEM beyond just IND-CCA. We call this attack a re-encapsulation attack, as it relies on the adversary encapsulating keying material that it previously obtained from decapsulation, causing two ciphertexts to decapsulate to the same key. We also show how our novel properties can prove the absence of such attacks.

Overall, our main contribution is the introduction of a class of novel binding properties. KEMs that satisfy these properties will leave fewer pitfalls for protocol designers, much in the same way as the related – but weaker – robustness notions do. Furthermore, our properties ensure the absence of, for example, several forms of misbinding attacks, notably including so-called re-encapsulation attacks. If we regard KEMs as one-pass key exchanges, our key-binding properties correspond to implicit key agreement properties.

2 Background

We first give the necessary background knowledge to understand the remainder of the paper. We begin with an introduction of KEMs in the computational model. Then, we discuss related work by Grubbs et al. [17].

A key-encapsulation scheme KEM consists of the three algorithms (KeyGen, Encaps, Decaps). It is associated with a key space $\mathcal{K}$ and a ciphertext space $\mathcal{C}$. The probabilistic key-generation algorithm KeyGen creates a key pair (pk, sk) where pk is the public key and sk is the secret key. Given a public key pk as input, the probabilistic encapsulation algorithm Encaps returns a ciphertext $c \in \mathcal{C}$ and a key $k \in \mathcal{K}$. To avoid ambiguity, we refer to k as the *output key* or the *shared secret.* The deterministic decapsulation algorithm Decaps uses a public key pk, a secret key sk and a ciphertext $c \in \mathcal{C}$ to compute an output key $k \in \mathcal{K}$ or the error symbol $\perp$ which represents rejection. If decapsulation cannot return $\perp$, we call KEM an *implicit rejection* KEM. Otherwise, we call it an *explicit rejection* KEM.

We write $\mathsf{Encaps}(pk; r)$ to make the randomness r that probabilistic algorithms like Encaps use explicit. We say that a KEM is ϵ-correct if for all $(sk, pk) \leftarrow \mathsf{KeyGen}()$ and $(c, k) \leftarrow \mathsf{Encaps}(pk; r)$, it holds that

$$\mathbf{Pr}[\mathsf{Decaps}(sk, c) \neq k] \leq \epsilon.$$

The security of a KEM is defined through indistinguishability of the output key $k \in \mathcal{K}$ computed by Encaps against different adversaries. The standard security notion is resistance against a chosen-ciphertext attack (IND-CCA) [8, 26]. We now recall the formal definition of the IND-CCA experiment shown in Figure 1.

In the experiment, we first create a key pair (sk, pk), sample randomness r, and then encapsulate against the public key $(c_0, k_0) \leftarrow \mathsf{Encaps}(pk; r)$. Then, we sample a random key k_1 from the key space and a random bit b. We give the adversary $\mathcal{A}$ access to c_0, k_b, and pk. The adversary then outputs a bit b' which indicates that it believes it received $k_{b'}$. Finally, the adversary wins if they correctly guessed b. The adversary has access to the decapsulation oracle $\mathsf{Decaps}(sk, \cdot)$, which returns the decapsulation of any ciphertext c that is not equal to the challenge ciphertext c_0.

$\text{IND-CCA}_{\mathcal{A}}^{\mathsf{KEM}}$:	$D(sk, pk, c)$:
$(sk, pk) \leftarrow\!\!\$\ \mathsf{KeyGen}()$	**if** $c \neq c_0$ **then**
$r \leftarrow\!\!\$\ \{0,1\}^*$	$k \leftarrow \mathsf{Decaps}(ct, sk)$
$(c_0, k_0) \leftarrow\!\!\$\ \mathsf{Encaps}(pk; r)$	**return** k
$k_1 \leftarrow\!\!\$\ \{0,1\}^*$	
$b \leftarrow\!\!\$\ \{0,1\}$	
$b' \leftarrow\!\!\$\ \mathcal{A}^{D(sk,pk,\cdot)}(c_0, k_b, pk)$	
return $b = b'$	

Figure 1: IND-CCA experiment.

In [17], Grubbs, Maram, and Paterson define anonymity, robustness, and so-called *collision freeness* for KEMs. We show these properties in Figure 7. Their work builds upon Mohassel's work [23] that studied these notions for KEMs built from PKEs, and only defined these properties for PKEs. Grubbs et al. investigate whether a PKE constructed via the KEM-DEM paradigm inherits anonymity and robustness from the underlying KEM. They show that this is indeed the case for explicitly rejecting KEMs. However, for implicitly rejecting KEMs, this is not the case in general. Since all NIST PQC finalist KEMs are implicitly rejecting KEMs constructed via variants of the FO transform, they then go on to analyze how the FO transform lifts robustness and anonymity properties from a PKE scheme, first to the KEM built via the FO transform, and then to the hybrid PKE scheme obtained via the KEM-DEM paradigm. Finally, they apply their generic analysis of the FO transform to the NIST PQC finalists Saber [14], Kyber [8], and Classic McEliece [5] as well as the NIST alternate candidate FrodoKEM [7]. For results of the specific schemes, we refer the reader to [17]. However, we want to a mention another finding of Grubbs et al. regarding the Classic McEliece scheme: for any plaintext m, they find that it is possible to construct a single ciphertext c which always decrypts to m under *any* Classic McEliece private key. Like them, we want to highlight that this does not indicate any problem with the IND-CCA security of Classic McEliece.

3 Re-encapsulation attacks

To further motivate the need for binding properties, we show a so-called *re-encapsulation attack*. To explain this, we use an authenticated key exchange protocol from the Kyber documentation, displayed in Figure 2. This protocol aims to establish a shared symmetric key between two parties that start only knowing eachother's public key.

We stress here that when the key exchange protocol is instantiated with Kyber, the protocol seems secure. However, can Kyber be replaced by any other KEM? Or, phrased differently, does this protocol solely depend on the core KEM property of IND-CCA? It turns out this is not the case.

To show this, we consider the same key exchange protocol, but instantiated with $\mathsf{KEM}_m^{\perp}$ from [18]. $\mathsf{KEM}_m^{\perp}$ is a so-called *FO-KEM*: a KEM created from an underlying PKE via the Fujisaki-Okamoto (FO) transform [15]. In a nutshell, the FO transform can be used to turn any weakly secure (i.e., IND-CPA) public-key encryption scheme into a strongly (i.e., IND-CCA) secure KEM scheme. Since the FO transform gives cryptographers a straightforward way to create a post-quantum secure KEM from a post-quantum secure PKE, these KEMs have surged in popularity, and are now the de-facto standard post-quantum secure KEMs. The NIST PQC process witnesses this as all KEM finalist are FO-KEMs.

In the context of our work, FO-KEMs are interesting because they have another property that is not captured by the current syntax of KEMs: a malicious party can learn m in addition to k when decapsulating. To understand why this is the case, we refer the reader to Figure 3 which shows $\mathsf{KEM}_m^{\perp}$ from [18]. From the graphic, we learn that the randomness r is only used to sample a random message m from the message space of the underlying PKE when computing Encaps. Since the party that computes Decaps computes m in the process, we argue that, when this party is malicious, they can learn m in addition to k. To capture this intuition with our KEM syntax, we write $(k, r) \leftarrow \mathsf{KEM.Decaps}(sk, c)$. Since m is the only value that depends on r, we believe that this faithfully captures an adversary learning m in addition to k when they compute Decaps of a FO-KEM.

We will now explain the re-encapsulation attack shown in Figure 4. In the attack, A and C are honest. The adversary B wants to coerce C into establishing a key shared with A, where C mistakenly assumes that A thinks they share the key with C: instead, A will think they share it with B. This is a so-called

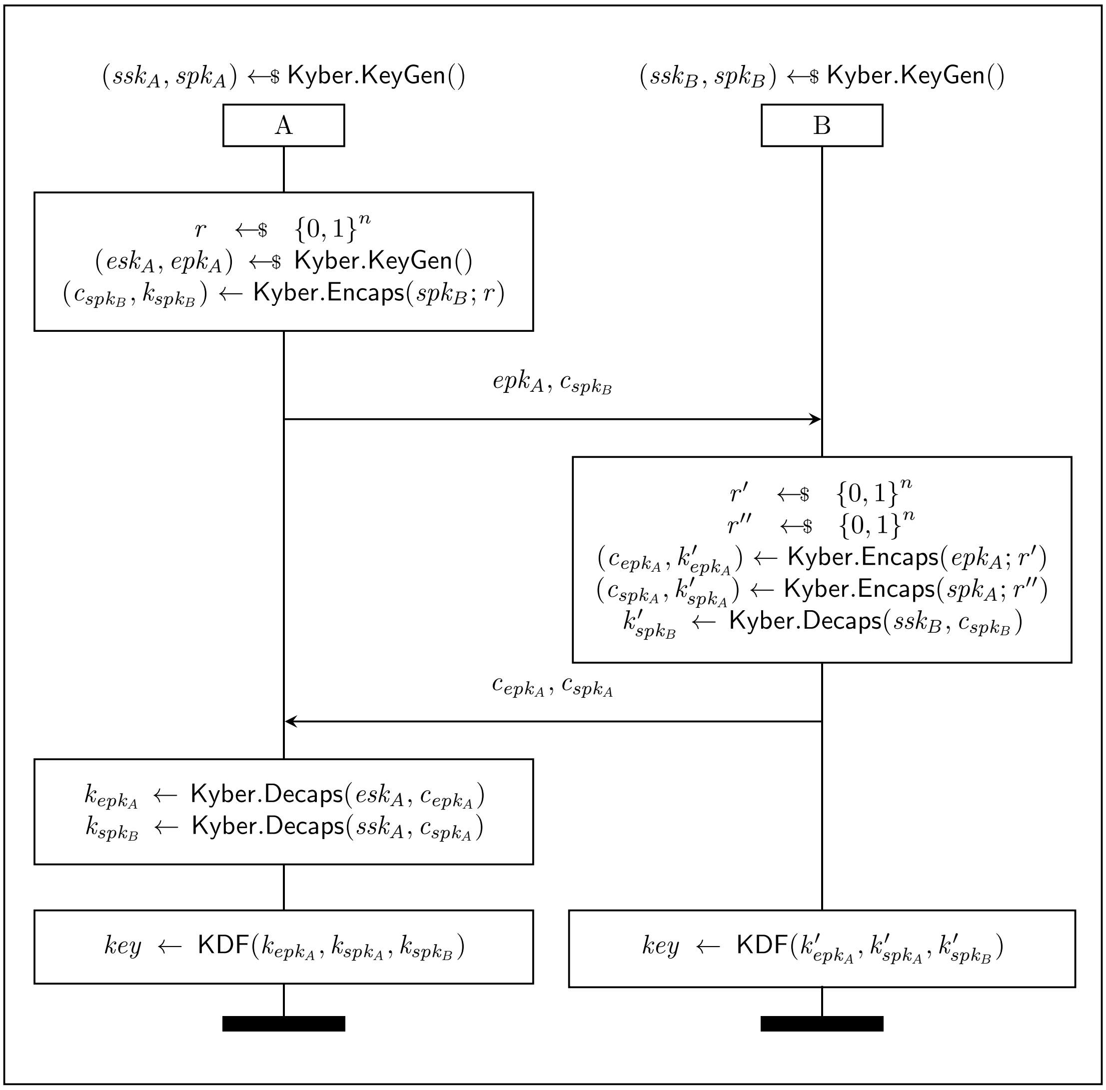

Figure 2: The authenticated key exchange suggested by Kyber [8].

$\mathsf{Encaps}(pk; r)$	$\mathsf{Decaps}(sk, ct)$
$m \leftarrow_r \mathcal{M}$	$m \leftarrow \mathsf{PKE.Dec}(sk, ct)$
$ct \leftarrow \mathsf{PKE.Enc}(pk, m; \mathsf{G}(m))$	**if** $m = \perp$ **then**
$k \leftarrow \mathsf{H}(m)$	**return** $\perp$
return (ct, k)	**else return** $k \leftarrow \mathsf{H}(m)$

Figure 3: IND-CCA secure key encapsulation mechanism $\mathsf{KEM}_m^{\perp}$ from [18].

unknown-key share attack [6] which violates the implicit key agreement of the authenticated key exchange from C's perspective, since C's session is intended to be with the honest A.

The attack proceeds as follows. When A initiates communication with B, B will decapsulate the ciphertext c_{spk_B} to obtain k_{r_0} and, more importantly, r_0 which was used by A to create c_{spk_B}. Now, B will impersonate A towards C by encapsulating against their static public key with r_0 and forwarding the resulting ciphertext and epk_A. Then, C will follow the protocol in a benign way, and respond with the expected values to A, as they think they are communicating with them. Finally, A will decapsulate the ciphertexts received from C, and both A and C will derive the final key. Since we instantiated the protocol with $\mathsf{KEM}_m^{\perp}$, the keys obtained via Decaps only depend on the randomness supplied by the encapsulating party. As a result, A and C derive the same *key*; this is a violation of implicit authentication since A thinks they now share a key with B, which does not match C's expectations.

One might wonder whether existing binding properties like SROB, and SCFR [17] (see Figure 7) are strong enough to prevent this attack. Unfortunately, this is not the case. The reason for this is that both properties reason about *a single ciphertext c* that should not decapsulate under different key pairs to the same key. However, our re-encapsulation attack revolves around two different ciphertexts: based on A's ciphertext, the malicious B creates a different ciphertext c_{spk_C} that decapsulates to the same key as c_{spk_B} by reusing the randomness r_0. In particular, if we use a robust PKE to instantiate the $\mathsf{KEM}_m^{\perp}$ in our attack example, the KEM is SROB (and thus SCFR), and still enables the attack. For more details, we refer the reader to Proposition 5.10 and Proposition 5.11. Thus, SROB and SCFR fail to capture our re-encapsulation attack.

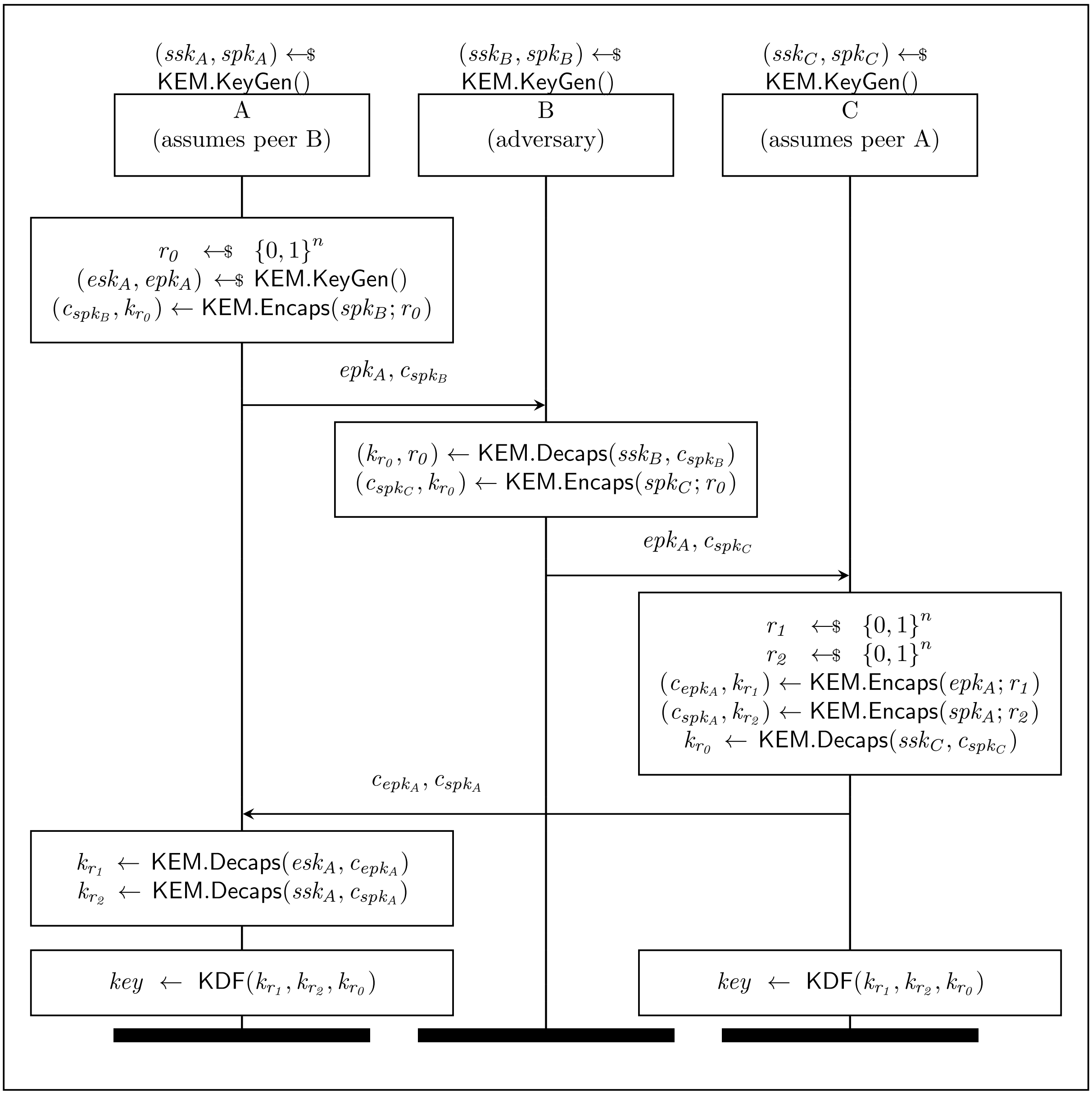

Figure 4: Re-encapsulation attack against the AKE suggested for the Kyber KEM [8] where the adversary B coerces honest A into unknowingly sharing the *key* with honest C, who thinks they are being contacted by A. This violates the implicit key agreement guarantee for C, who expects to share a key with someone that assumes C is the peer. At the AKE protocol level, this corresponds to an unknown-key share attack. Note that this attack is only possible when the AKE is instantiated with a FO-KEM that does not bind the output key to the public key, and is not possible when instantiated with Kyber.

4 New security notions for KEMs

To establish a generic family of binding properties of KEMs, we first identify the elements that may be candidates for binding. The syntax of a KEM includes a long-term key pair, a ciphertext, and an (output) key. In some formalizations, the randomness of the KEM is made explicit, but we are looking for universal black-box notions that do not require us to know the internals of a KEM. With respect to the long-term keypair, we note that we want the guarantees to be relevant for both sender and recipient, which means we will only consider the public key as the identifying aspect of the keypair. This leaves us with pk, ct, and k: we expect that for each invocation of the KEM's encapsulation with the same pk, the outputs ct and k would be unique.

We can thus wonder: if we have a specific instance of one of these, does that mean the others are uniquely determined? If I have a given ciphertext, can it only be decapsulated by one key?

4.1 Design choices

To define our class of properties, we make the following design decisions:

1. We consider the set of potential binding elements $BE = \{\mathsf{pk}, \mathsf{ct}, \mathsf{k}\}$.

2. We will consider if an instance of a set $P \subset BE$ "binds" some instance of another set of elements $Q \subset BE$ with respect to decapsulation with the KEM. Thus, "P binds Q" if for fixed instances of P there are no collisions in the instances of Q.

3. Given that pk is re-used, it does not bind any values, and we hence exclude it from occurring in P. However, ciphertexts or keys might bind a public key pk, so it may occur in Q.

4. Adding multiple elements in the set Q corresponds to a logical "and" of the singleton versions, i.e., we have that P binds $\{q_1, \ldots, q_n\}$ iff for all $i \in [n]$. P binds $\{q_i\}$. We therefore choose to not enumerate all logical combinations into separate properties, and choose to model the core properties with $|Q| = 1$.

5. We require P and Q to be disjoint: elements that would occur on both sides are trivially bound.

6. For all of our properties, we will consider honest variants (i.e., the involved key pairs are output by the key generation algorithm of the KEM) and malicious variants (i.e., the adversary can create the key pairs any way they like in addition to the key generation).

Based on the above choices, we have three choices for P: $\{\mathsf{k}\}$, $\{\mathsf{ct}\}$, $\{\mathsf{k}, \mathsf{ct}\}$. For Q, we can choose from $\{\mathsf{pk}\}$, $\{\mathsf{k}\}$, and $\{\mathsf{ct}\}$. Without disjointness this would yield 3×3 options, but since we require the sets to be disjoint, this yields the 5 combinations that we will show in Figure 1.

4.2 Naming conventions

Naming security notions is hard; once names are fixed, they tend to stick around for (too) long. We opt here for clarity and being descriptive at the cost of some verbosity. In the literature, it is more common to collapse all of these properties into "robustness" or "collision-freeness", but this becomes very ambiguous because one can imagine many subtle variants, depending on the exact robust/collision-free element in the construction. This has lead to a long list of non-descriptive names in the literature, including: Robustness, Fuller Fobustness (FROB), even Fuller Robustness (eFROB), SROB, KROB, XROB, CROB, WROB, SCFR, WCFR, etc. In contrast, we illustrate our descriptive naming scheme for our binding properties in Figure 5.

4.3 Generic security notion for binding properties of KEMs

We now introduce the generic security notion for our class of binding properties. In Figure 6 we show the corresponding generic game.

Definition 4.1. *Let* KEM *be a KEM. Furthermore, let* $X \in \{HON, MAL\}$*, let* $P \in \{\{\mathsf{k}\}, \{\mathsf{ct}\}, \{\mathsf{k}, \mathsf{ct}\}\}$*, and let* $Q \in \{\{\mathsf{pk}\}, \{\mathsf{k}\}, \{\mathsf{ct}\}\}$ *such that* $P \cap Q = \emptyset$*.*

We say that KEM *is* X*-BIND-*P*-*Q*-secure iff for any PPT adversary* $\mathcal{A}$*, the probability that* $X\text{-}BIND\text{-}P\text{-}Q_{\mathcal{A}}^{\mathsf{KEM}}$ *returns 1 (true) is negligible.*

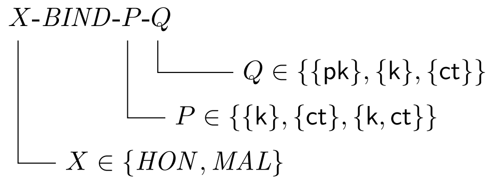

Figure 5: Design space and naming conventions for our security properties: For a KEM scheme that is *X-BIND-P-Q* secure, we say that "P [honestly|maliciously] binds Q", using "honestly" when $X = HON$ and "maliciously" when $X = MAL$. Alternative wording can be "Given P, Q is [honestly|maliciously] collision-free". We commonly omit set brackets in the notation when clear from the context, and use uppercase for all characters. For example, *HON-BIND-CT-PK* corresponds to "ct honestly binds pk."

In our definitions, $X \in \{HON, MAL\}$ indicates the adversary's control over the considered key pairs. In the honest ($X = HON$) case, two honestly generated key pairs are considered in the game, whereas in the malicious ($X = MAL$) case, the adversary can choose or construct the key pairs in any way they want. In the honest case, we give the adversary access to a decapsulation oracle $D_b(\mathsf{sk}_b, \cdot)$, where $b \in \{0, 1\}$, that they can use the decapsulate ciphertexts with either secret key. This is not necessary for the malicious case since the adversary knows the secret keys.

$X\text{-}BIND\text{-}P\text{-}Q_{\mathcal{A}}^{\mathsf{KEM}}$:

- **if** $X = HON$:
  - $\mathsf{sk}_0, \mathsf{pk}_0 \leftarrow \mathsf{KeyGen}()$
  - $\mathsf{sk}_1, \mathsf{pk}_1 \leftarrow \mathsf{KeyGen}()$
  - $\mathsf{ct}_0, \mathsf{ct}_1 \leftarrow \mathcal{A}^{D_b(\mathsf{sk}_b, \cdot)}(\mathsf{pk}_0, \mathsf{pk}_1)$
- **if** $X = MAL$:
  - $(\mathsf{sk}_0, \mathsf{pk}_0), (\mathsf{sk}_1, \mathsf{pk}_1), \mathsf{ct}_0, \mathsf{ct}_1 \leftarrow \mathcal{A}()$
- $\mathsf{k}_0 \leftarrow \mathsf{KEM.Decaps}(\mathsf{sk}_0, \mathsf{pk}_0, \mathsf{ct}_0)$
- $\mathsf{k}_1 \leftarrow \mathsf{KEM.Decaps}(\mathsf{sk}_1, \mathsf{pk}_1, \mathsf{ct}_1)$
- **if** $k_0 = \bot \lor k_1 = \bot$: **return** 0
- // $\mathcal{A}$ wins if $\neg\big((\forall x \in P \,.\, x_0 = x_1) \implies (\forall y \in Q \,.\, y_0 = y_1)\big)$
- **return** $\forall x \in P \,.\, x_0 = x_1 \land \exists y \in Q \,.\, y_0 \neq y_1$

Figure 6: Generic game for our new binding notions *X-BIND-P-Q*.

4.4 Relation between binding and contributive behavior

In the context of other cryptographic primitives, the notion of contributive (or contributory) behavior exists. Intuitively, in a two-party protocol that yields some randomized output, a protocol is contributive if the output is not only determined by one of the parties, but both contribute to the results.

For example, in a standard FO-KEM such as $\mathsf{KEM}_m^{\bot}$ from [18], the randomness sampled for encapsulation is the direct (and only) input for the key derivation function (KDF). Thus, for $\mathsf{KEM}_m^{\bot}$, the only party that contributes to the output key is the sender. We say that such KEMs are non-contributory, and can enable re-encapsulation attacks, as described in Section 3.

In contrast, if the KEM's key binds the public key (e.g., by including the public key in the KDF) then the KEM satisfies *MAL-BIND-K-PK*, and we say that the KEM is contributory because the recipients' key contributes to the output key.

When the KEM's key binds the ciphertext then the KEM satisfies *MAL-BIND-K-CT*. However, it is not immediately clear whether this is enough to make the KEM contributory, and it depends on the collision freeness [17] (SCFR) of the underlying PKE. If the underlying PKE is not strongly collision free, that is, it is possible to decrypt a single ciphertext to the same message with different secret keys, then the KEM is not contributory. The reason for that is that a single ciphertext is valid for multiple public keys, and thus the identity of the receiver is not bound by including the ciphertext in the output key of the KEM. On the other hand, if the PKE is strongly collision free (or even robust) then including the ciphertext makes the KEM contributory. See Theorem 6.1, and Section 6 for more details.

ID	P	Q	Property	Explanation
1	$\{\mathsf{k}\}$	$\{\mathsf{ct}\}$	*X-BIND-K-CT*	Output key binds the ciphertext.
2	$\{\mathsf{k}\}$	$\{\mathsf{pk}\}$	*X-BIND-K-PK*	Output key binds the public key.
3	$\{\mathsf{ct}\}$	$\{\mathsf{k}\}$	*X-BIND-CT-K*	Ciphertext binds the output key.
4	$\{\mathsf{ct}\}$	$\{\mathsf{pk}\}$	*X-BIND-CT-PK*	Ciphertext binds the public key. The honest notion *HON-BIND-CT-PK* is equivalent to SROB from [17]
5	$\{\mathsf{k}, \mathsf{ct}\}$	$\{\mathsf{pk}\}$	*X-BIND-K, CT-PK*	Together, the output key and ciphertext bind the public key. The honest notion *HON-BIND-K, CT-PK* is equivalent to SCFR from [17].

Table 1: This table contains the five core instantiations of our generic binding property *X-BIND-P-Q* before choosing $X \in \{HON, MAL\}$.

```
SROB-CCA_A^KEM :
  (sk_0, pk_0) ← KeyGen()
  (sk_1, pk_1) ← KeyGen()
  ct ← A^{D(·,·)}(pk_0, pk_1)
  k_0 ← KEM.Decaps(sk_0, pk_0, ct)
  k_1 ← KEM.Decaps(sk_1, pk_1, ct)
  return k_0 ≠ ⊥ ∧ k_0 ≠ ⊥

SCFR-CCA_A^KEM :
  (sk_0, pk_0) ← KeyGen()
  (sk_1, pk_1) ← KeyGen()
  ct ← A^{D(·,·)}(pk_0, pk_1)
  k_0 ← KEM.Decaps(sk_0, pk_0, ct)
  k_1 ← KEM.Decaps(sk_1, pk_1, ct)
  return k_0 = k_0 ≠ ⊥

HON-BIND-CT-PK_A^KEM :
  sk_0, pk_0 ← KeyGen()
  sk_1, pk_1 ← KeyGen()
  ct ← A^{D_b(sk_b,·)}(pk_0, pk_1)
  k_0 ← KEM.Decaps(sk_0, pk_0, ct)
  k_1 ← KEM.Decaps(sk_1, pk_1, ct)
  if k_0 = ⊥ ∨ k_1 = ⊥
    return 0
  return pk_0 ≠ pk_1

HON-BIND-K,CT-PK_A^KEM :
  sk_0, pk_0 ← KeyGen()
  sk_1, pk_1 ← KeyGen()
  ct ← A^{D_b(sk_b,·)}(pk_0, pk_1)
  k_0 ← KEM.Decaps(sk_0, pk_0, ct)
  k_1 ← KEM.Decaps(sk_1, pk_1, ct)
  if k_0 = ⊥ ∨ k_1 = ⊥
    return 0
  return k_0 = k_1 ∧ pk_0 ≠ pk_1

MAL-BIND-CT-PK_A^KEM :
  (sk_0, pk_0), (sk_1, pk_1), ct ← A()
  k_0 ← KEM.Decaps(sk_0, pk_0, ct_0)
  k_1 ← KEM.Decaps(sk_1, pk_1, ct_1)
  if k_0 = ⊥ ∨ k_1 = ⊥
    return 0
  return pk_0 ≠ pk_1

MAL-BIND-K,CT-PK_A^KEM :
  (sk_0, pk_0), (sk_1, pk_1), ct ← A()
  k_0 ← KEM.Decaps(sk_0, pk_0, ct)
  k_1 ← KEM.Decaps(sk_1, pk_1, ct)
  if k_0 = ⊥ ∨ k_1 = ⊥
    return 0
  return k_0 = k_1 ∧ pk_0 ≠ pk_1
```

Figure 7: At the top, the strong robustness and strong collision freeness definitions from [17]. In the middle, our *HON-BIND-CT-PK* and *HON-BIND-K, CT-PK* definitions which correspond to SROB and SCFR respectively. At the bottom, our *MAL-BIND-CT-PK* and *MAL-BIND-K, CT-PK* definitions which give the adversary control over the key generation.

5 Relations and implications

In this section, we establish the relations between the various notions. In general, as we show in Section 5.1, it turns out that the properties are largely orthogonal: there exist KEMs that have some of these properties, but do not meet other properties in our hierarchy.

However, in practice, many KEM designs come from more narrow subclasses, in which some of these properties collapse or become unsatisfiable. For example, for implicitly rejecting KEMs, ciphertexts do not bind any other values. In Section 5.2 we show the resulting reduced hierarchy for implicitly rejecting KEMs.

5.1 General relations for any KEM

In this section, we first prove some generic properties of our generic binding definition, then we go on to clarify the relations between the binding properties from Table 1 by giving a separating example for each pair. Table 2 shows a summary.

Our main outcome is the hierarchy of properties that we present at the end in Figure 8.

5.1.1 General implications

Lemma 5.1. *Let* KEM = (KeyGen, Encaps, Decaps) *be a key encapsulation mechanism. If* KEM *is MAL-BIND-P-Q-secure, then* KEM *is also HON-BIND-P-Q-secure.*

Proof. We prove the contraposition. Let $\mathcal{A}$ be an attacker against $\textit{HON-BIND-P-Q}_{\mathcal{A}}^{\mathsf{KEM}}$. We construct attacker $\mathcal{B}$ against $\textit{MAL-BIND-P-Q}_{\mathcal{B}}^{\mathsf{KEM}}$. $\mathcal{B}$ generates two key pairs honestly, and then calls $\mathcal{A}$ with these key pairs to win $\textit{MAL-BIND-P-Q}_{\mathcal{B}}^{\mathsf{KEM}}$ with the same non-negligible probability that $\mathcal{A}$ wins $\textit{HON-BIND-P-Q}_{\mathcal{A}}^{\mathsf{KEM}}$ with. □

Lemma 5.2. *Let* KEM = (KeyGen, Encaps, Decaps) *be a key encapsulation mechanism. For* $X \in \{MAL, HON\}$*, if* KEM *is X-BIND-P-Q′-secure and* $P \subseteq P' \wedge Q \subseteq Q'$*, then* KEM *is also X-BIND-P′-Q-secure.*

Proof. Assume KEM is *X-BIND-P-Q′*-secure and $P \subseteq P' \wedge Q \subseteq Q'$. Now assume towards contradiction that KEM is not *X-BIND-P′-Q*-secure. Thus, there exists an attacker $\mathcal{A}$ that wins the $\textit{X-BIND-P'-Q}_{\mathcal{A}}^{\mathsf{KEM}}$ game with non-negligible probability. From $\mathcal{A}$, we now build an attacker $\mathcal{B}$ that wins the $\textit{X-BIND-P-Q'}_{\mathcal{B}}^{\mathsf{KEM}}$ game with non-negligible probability. $\mathcal{B}$ simply calls $\mathcal{A}$ to win $\textit{X-BIND-P-Q'}_{\mathcal{B}}^{\mathsf{KEM}}$ with the same probability that $\mathcal{A}$ wins $\textit{X-BIND-P'-Q}_{\mathcal{A}}^{\mathsf{KEM}}$. We now argue why the *winning condition*

$$\forall x \in P' \,.\, x_0 = x_1 \wedge \exists y \in Q \,.\, y_0 \neq y_1$$

of $\mathcal{A}$ in $\textit{X-BIND-P'-Q}_{\mathcal{A}}^{\mathsf{KEM}}$ implies the winning condition

$$\forall x \in P \,.\, x_0 = x_1 \wedge \exists y' \in Q' \,.\, y'_0 \neq y'_1$$

of $\mathcal{B}$ in $\textit{X-BIND-P-Q'}_{\mathcal{B}}^{\mathsf{KEM}}$. Since $P \subseteq P'$, each equality constraint in P is also present in P', and thus satisfied. Since $Q \subseteq Q'$, we can we can use the witness y from Q as witness y' for Q' to satisfy the existential constraint.

□

5.1.2 Separating examples

Proposition 5.3. *There exists a KEM scheme that is MAL-BIND-K-CT but not HON-BIND-K-PK.*

Proof. This KEM scheme is the Classic McEliece scheme from [5]. Classic McEliece derives the output key as follows: $\mathsf{k} \leftarrow H(m||c||H'(m))$. Here, m is a message from the message space of the underlying PKE, c is the ciphertext created by encapsulating m with a public key, and H, H' are hash functions. Thus, Classic McEliece is *MAL-BIND-K-CT*.

From [17], we know that it is possible to construct a single ciphertext c that decrypts to the same message under any Classic McEliece private key. Therefore, Classic McEliece is not *HON-BIND-K-PK* because we can use c to derive the same output key k under any key pair. In fact, Classic McEliece is not *X-BIND-CT-PK* for the same reason. □

Proposition 5.4. *There exists a KEM scheme that is MAL-BIND-K-CT but not HON-BIND-CT-PK.*

Proof. See Proposition 5.3. □

Proposition 5.5. *There exists a KEM scheme that is MAL-BIND-K-CT but not HON-BIND-CT-K.*

Proof. We propose a variant of the Classic McEliece scheme where

$$\mathsf{KEM.Decaps}(sk, pk, ct) = k \leftarrow \mathsf{ClassicMcEliece.Decaps}(sk, pk, ct);\ \text{return } \mathsf{H}(k||pk).$$

For KEM.Encaps, we make the same changes to the key derivation. Otherwise, KEM behaves the same. Observe that this scheme is still *MAL-BIND-K-CT* because the ciphertext is hashed into the output key. Additionally, this scheme is also *MAL-BIND-K-PK* since it hashes the public key into the output key.

We create a ciphertext c that decrypts to the same m for any key pair, as described in [17]. Normally, c would decapsulate to the same output key under every key pair because only c and the underlying message m are hashed into the output key. However, with our changes to the decapsulation, KEM produces different output keys for different Classic McEliece key pairs. Thus, our variant is not *HON-BIND-CT-K*. □

Proposition 5.6. *There exists a KEM scheme that is MAL-BIND-K-CT but not HON-BIND-K, CT-PK.*

Proof. Again, we use the Classic McEliece scheme from [5]. As previously noted, Classic McEliece is *MAL-BIND-K-CT*.

Analogous to Proposition 5.5, we create a ciphertext c that decrypts to the same m for any key pair, as described in [17]. Since Classic McEliece derives the output key only from the ciphertext c and the message m, c decapsulates to the same output key for every secret key. Thus, Classic McEliece is not *X-BIND-K, CT-PK*. □

Proposition 5.7. *There exists a KEM scheme that is MAL-BIND-K-PK but not HON-BIND-K-CT.*

Proof. We propose a variant KEM of the $\mathsf{KEM}_m^\perp$ scheme shown in Figure 3 where encapsulation is defined as follows:

$$\mathsf{KEM.Encaps}(pk; r) = (ct, k) \leftarrow \mathsf{KEM}_m^\perp\mathsf{.Encaps}(pk; r \text{ div } 2;\ \text{return } (ct, \mathsf{H}(k||pk)).$$

In a nutshell, we add the public key of the recipient to the key derivation, and shorten r by one bit. Decapsulation is defined as follows:

$$\mathsf{KEM.Decaps}(sk, pk, ct) = k \leftarrow \mathsf{KEM}_m^\perp\mathsf{.Decaps}(sk, pk, ct);\ \text{return } \mathsf{H}(k||pk).$$

Note that KEM is *MAL-BIND-K-PK* because it hashes pk into the output key. However, KEM is not *HON-BIND-K-CT* encapsulating against the same public key with the two values r_1, r_2, which only differ in the last bit, results in the same ciphertext, and the same output key. □

Proposition 5.8. *There exists a KEM scheme that is MAL-BIND-K-PK but not HON-BIND-CT-PK.*

Proof. We propose a variant of the Classic McEliece scheme that replaces the the ciphertext in the key derivation with the public key like this $\mathsf{k} \leftarrow \mathsf{H}(m||pk||H'(m))$. Otherwise, the scheme behaves the same. Observe that this scheme is no longer *MAL-BIND-K-CT* because the ciphertext is no longer hashed into the output key. However, this scheme is now *MAL-BIND-K-PK* since it hashes the public key into the output key.

Again, we create a ciphertext c that decrypts to the same m for any key pair [17]. Since c decapsulates under any private key, our variant is not *HON-BIND-CT-PK*. □

Proposition 5.9. *There exists a KEM scheme that is MAL-BIND-K-PK but not HON-BIND-CT-K.*

Proof. See Proposition 5.5. □

Proposition 5.10. *There exists a KEM scheme that is MAL-BIND-CT-PK but not HON-BIND-K-CT.*

Proof. (Sketch) We conjecture that $\mathsf{KEM}_m^\perp$ from Figure 3 is *MAL-BIND-CT-PK* when the underlying PKE is robust. We next observe that in this KEM scheme the output key k only depends on the randomness supplied to Encaps. Thus, we can create multiple ciphertexts that decapsulate to the same output key k for honestly generated key pairs by re-using the randomness. For an example, see Figure 4. Therefore, the $\mathsf{KEM}_m^\perp$ is not *HON-BIND-K-CT*. In fact, it is also not *HON-BIND-K-PK*. □

Proposition 5.11. *There exists a KEM scheme that is MAL-BIND-CT-PK but not HON-BIND-K-PK.*

Proof. See Proposition 5.10. □

Proposition 5.12. *There exists a KEM scheme that is MAL-BIND-CT-K but not HON-BIND-K-CT.*

Proof. (Sketch) We conjecture that $\mathsf{KEM}_m^{\perp}$ from Figure 3 is *MAL-BIND-CT-K* when the underlying PKE is robust. The remaining argument is analogous to Proposition 5.10. □

Proposition 5.13. *There exists a KEM scheme that is MAL-BIND-CT-K but not HON-BIND-K-PK.*

Proof. (Sketch) We conjecture that $\mathsf{KEM}_m^{\perp}$ from Figure 3 is *MAL-BIND-CT-K* when the underlying PKE is robust. The remaining argument is analogous to Proposition 5.10. □

Proposition 5.14. *There exists a KEM scheme that is MAL-BIND-CT-K but not HON-BIND-CT-PK.*

Proof. To argue for this proposition, we supply an artificial construction. Assume the KEM scheme is *X-BIND-CT-K*. This KEM scheme, however, has a backdoor value m' with the following behavior:

$$\begin{aligned}&\mathsf{Encaps}(m', pk_x) = c\\&\mathsf{Decaps}(sk_x, pk_x, c) = m'\\&k \leftarrow H(m')\end{aligned}$$

for (sk_x, pk_x) being all possible key pairs and c being a constant. In this scenario, *X-BIND-CT-K* would still hold by construction. However, for different public keys the attacker is now able to produce the same ciphertext using the backdoor value m'. Hence, the KEM does not offer *X-BIND-CT-PK*. □

Proposition 5.15. *There exists a KEM scheme that is MAL-BIND-CT-K but not HON-BIND-K,CT-PK.*

Proof. Let $\mathsf{KEM} = (\mathsf{KeyGen}, \mathsf{Encaps}, \mathsf{Decaps})$ be a *MAL-BIND-CT-K* KEM. We now construct a new KEM KEM' from KEM that is *MAL-BIND-CT-K* but not *HON-BIND-K,CT-PK*. Let $\mathsf{KeyGen}' =$

$$(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KeyGen}; \mathsf{pk}' \leftarrow \mathsf{pk} \cdot 2 + b \leftarrow\!\!\$ \{0,1\}; \text{return } (\mathsf{sk}, \mathsf{pk}').$$

Let $\mathsf{Decaps}(\mathsf{sk}, \mathsf{pk}, \mathsf{ct})' =$

$$\mathsf{Decaps}(\mathsf{sk}, \mathsf{pk} \text{ div } 2, \mathsf{ct}).$$

Let $\mathsf{Encaps}(\mathsf{pk}; r)' =$

$$\mathsf{Encaps}(\mathsf{pk} \text{ div } 2; r).$$

KEM' is still *MAL-BIND-CT-K* as we can use an attacker $\mathcal{A}'$ against KEM' to built an attacker $\mathcal{A}$ against KEM. However, KEM' is not *HON-BIND-K,CT-PK* as we have two public keys for each ciphertext that yield the same output key. □

Proposition 5.16. *There exists a KEM scheme that is MAL-BIND-K,CT-PK but not HON-BIND-K-CT.*

Proof. We have that *MAL-BIND-CT-PK* $\implies$ *MAL-BIND-K,CT-PK* by Lemma 5.2. Thus, the KEM from Proposition 5.10 is also *MAL-BIND-K,CT-PK* and not *HON-BIND-K-CT*. □

Proposition 5.17. *There exists a KEM scheme that is MAL-BIND-K,CT-PK but not HON-BIND-K-PK.*

Proof. We have that *MAL-BIND-CT-PK* $\implies$ *MAL-BIND-K,CT-PK* by Lemma 5.2. Thus, the KEM from Proposition 5.11 is also *MAL-BIND-K,CT-PK* and not *HON-BIND-K-PK*. □

Proposition 5.18. *There exists a KEM scheme that is MAL-BIND-K,CT-PK but not HON-BIND-CT-PK.*

Proof. From [17], we know that, for instance, Kyber is SCFR but not SROB. *MAL-BIND-K,CT-PK* corresponds to SCRF, and *HON-BIND-CT-PK* corresponds SROB in our notation. □

Proposition 5.19. *There exists a KEM scheme that is MAL-BIND-K,CT-PK but not HON-BIND-CT-K.*

Proof. Kyber is such a KEM scheme. From [17], we know that Kyber is SCFR, which corresponds to *HON-BIND-K,CT-PK* in our notation. Under the assumption that Kyber has no weak keys, it is thus also *MAL-BIND-K,CT-PK*. However, Kyber is not *HON-BIND-CT-K* because when decapsulation fails it includes a random, secret seed s that is part of the secret key in the implicit rejection key: $\mathsf{k} \leftarrow H(s, H(c))$. Here, H is a hash function, and c is a ciphertext. Thus, Kyber returns different rejection keys for different secret keys, and is not *HON-BIND-CT-K*. □

	¬(*HON-BIND-K-CT*)	¬(*HON-BIND-K-PK*)	¬(*HON-BIND-CT-PK*)	¬(*HON-BIND-CT-K*)	¬ (*HON-BIND-K,CT-PK*)
MAL-BIND-K-CT	X	Proposition 5.3	Proposition 5.4	Proposition 5.5	Proposition 5.6
MAL-BIND-K-PK	Proposition 5.7	X	Proposition 5.8	Proposition 5.9	X Lemma 5.2
MAL-BIND-CT-PK	Proposition 5.10	Proposition 5.11	X	X Lemma 5.20	X Lemma 5.2
MAL-BIND-CT-K	Proposition 5.12	Proposition 5.13	Proposition 5.14	X	Proposition 5.15
MAL-BIND-K,CT-PK	Proposition 5.16	Proposition 5.17	Proposition 5.18	Proposition 5.19	X

Table 2: Summary of the separating examples between our binding properties. A X means that a separating example does not exists.

Lemma 5.20. *Let* KEM *be a KEM that is HON-BIND-CT-PK secure. Then* KEM *is also HON-BIND-CT-K secure.*

Proof. Because a ciphertext of KEM binds (uniquely determines) the public key, and the keypairs are honestly generated this uniquely determines a private key with overwhelming probability. The deterministic nature of decapsulation then ensures binding of the output key, thus meeting *HON-BIND-CT-K*. □

We visualize the resulting hierarchy in Figure 8.

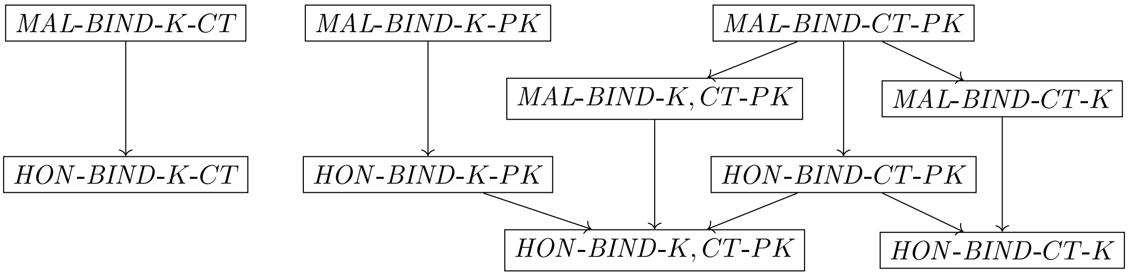

Figure 8: General hierarchy of binding properties for KEMs. An edge from A to B indicates that any KEM that is A-secure, is also B-secure. Missing edges represent the existence of separating examples.

5.2 Relations for implicitly-rejecting KEMs

Many real-world KEMs are so-called *implicitly rejecting*: their decapsulation algorithm never returns $\perp$ for a valid ciphertext and any valid private key. However, only when decapsulating with the correct private key (corresponding to the public key used for encapsulation), the correct key is output.

Intuitively, an implicitly rejecting KEM is similar to an implicitly authenticated key exchange: successfully completing the protocol does not imply that someone else has the same key, or sent any message; instead, the guarantee is that only the correct party can possibly compute the same secret key.

However, a trivial side effect of this is that such a KEM cannot be binding the ciphertext to any other value: any ciphertext will be accepted, and these will (with overwhelming probability) decapsulate to different keys. A special case of this is the observation in [17] that an implicitly-rejecting KEM cannot satisfy SROB, i.e., *HON-BIND-CT-PK*.

Theorem 5.21. *An implicitly-rejecting KEM* KEM *cannot satisfy X-BIND-CT-PK or X-BIND-CT-K for* $X \in \{HON, MAL\}$.

Proof. We now show that:

1. KEM cannot be *HON-BIND-CT-PK*-secure
2. KEM cannot be *HON-BIND-CT-K*-secure

The analogous statements for the malicious case then follow by the contraposition of Lemma 5.1.

1. We construct an attacker $\mathcal{A}$ against *HON-BIND-CT-PK*. As KEM is implicitly rejecting, Decaps will always return a value k. Hence $\mathcal{A}$ can choose arbitrary values for the ciphertext, and $\mathsf{Decaps}(sk_b, pk_b, c)$ will always return some k. Since Decaps returning a value that is not equal to $\perp$ is all $\mathcal{A}$ needs to achieve, $\mathcal{A}$ trivially wins the game *HON-BIND-CT-PK*.

2. The statement follows from Lemma 5.20 and the previous statement.

□

Theorem 5.22. *Let* KEM *be an implicitly rejecting KEM, and let* $X \in \{HON, MAL\}$*. We have that* KEM *is* X*-BIND-K-PK secure if and only if it is* X*-BIND-K,CT-PK secure.*

Proof. (Sketch) One direction of the implication follows immediately from Lemma 5.2. For the other direction, the underlying argument is that for implicitly rejecting KEMs, the ciphertext does not contribute to the binding, and only the key is relevant. □

Thus, for implicitly-rejecting KEMs, we have a reduced hierarchy of relevant properties. The separation between honest and malicious variants persists, but only two core properties are relevant and distinct, which are the key-binding properties. Since X-*BIND-K-PK* and X-*BIND-K,CT-PK* are equivalent for such KEMs, we choose to consider the more restrictive formulation X-*BIND-K-PK*.

This leaves us with a simple hierarchy with only four relevant binding properties overall for implicitly-rejecting KEMs, which we visualize in Figure 9.

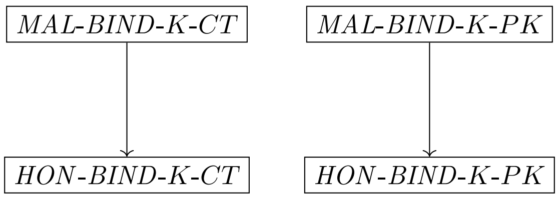

Figure 9: Restricted hierarchy of binding properties for implicitly-rejecting KEMs. An edge from A to B indicates that any KEM that is A-secure, is also B-secure. Missing edges represent the existence of separating examples.

6 Achieving strong binding properties

We give suggestions on how (not) to construct KEMs that achieve some of our binding properties.

6.1 Ensuring output keys bind to public keys or ciphertexts

In nearly all KEM designs, the last step of encapsulation and decapsulation is to produce the output key by using a KDF (Key Derivation Function); if not, such a step can be added. In order to ensure that the key binds another element, we can simply add this element to the KDF inputs.

Thus, to achieve *MAL-BIND-K-CT* and *MAL-BIND-K-PK*, we can simply add CT and PK to the input of the key derivation function.

Of course, this is not the only way to achieve such binding properties: Leaving out either CT or PK does not mean that the corresponding property does not hold: it simply means there is a proof obligation to show that a KEM meets such a binding property without this construction.

In practice, and in particular for post-quantum KEMs, the public key can be substantially larger than the ciphertext. It may therefore be desirable to avoid directly including the public key in the key derivation. For this case, we prove Theorem 6.1 below. It implies that KEM designers that want to achieve X-*BIND-K-PK* but want to avoid using the public key in the KDF, can instead use a robust PKE, and include the ciphertext in the KDF, which will yield the desired binding to the public key.

Theorem 6.1. *Let* KEM = (KeyGen, Encaps, Decaps) *be a key encapsulation mechanism. For all* $X \in \{MAL, HON\}$ *we have that if* KEM *is* X*-BIND-K-CT-secure and* X*-BIND-CT-PK-secure, then* KEM *is also* X*-BIND-K-PK-secure.*

Proof. We prove the contraposition. Assume there is an adversary $\mathcal{A}$ that wins against the $X\text{-}BIND\text{-}K\text{-}PK_{\mathcal{A}}^{\mathsf{KEM}}$ game with non-negligible probability γ. From $\mathcal{A}$, we construct an adversary $\mathcal{B}$ that wins the $X\text{-}BIND\text{-}K\text{-}CT_{\mathcal{A}}^{\mathsf{KEM}}$ game or the $X\text{-}BIND\text{-}CT\text{-}PK_{\mathcal{A}}^{\mathsf{KEM}}$ game with non-negligible probability. To construct $\mathcal{B}$, we use $\mathcal{A}$ without any modifications. Observe that $\mathcal{A}$ wins $X\text{-}BIND\text{-}K\text{-}PK_{\mathcal{A}}^{\mathsf{KEM}}$ if they can create two ciphertexts (c_0, c_1), s.t. $k_0 = \mathsf{Decaps}(sk_0, pk_0, c_0) = \mathsf{Decaps}(sk_1, pk_1, c_1) = k_1$ for two distinct key pairs (sk_0, pk_0) and (sk_1, pk_1). We do a case distinction on $c_0 = c_1$:

1. If $c_0 = c_1$ with probability $\rho \in [0, 1]$ then $\mathcal{A}$ wins the $X\text{-}BIND\text{-}CT\text{-}PK^{\mathsf{KEM}}_{\mathcal{A}}$ game with non-negligible probability $\gamma \cdot \rho$, because $\mathcal{A}$ can find a single ciphertext that decapsulates under two different public key pairs.

2. If $c_0 \neq c_1$ with probability $1 - \rho$ then $\mathcal{A}$ wins the $X\text{-}BIND\text{-}K\text{-}CT^{\mathsf{KEM}}_{\mathcal{A}}$ game with non-negligible probability $\gamma \cdot (1 - \rho)$, because $\mathcal{A}$ can find two different ciphertexts that decapsulate to the same output key.

□

6.2 Further considerations

Ensuring ciphertexts bind to public keys or output keys As we have seen, implicitly-rejecting KEMs cannot bind ciphertexts to any other value: these properties can only be achieved by explicitly-rejecting KEMs.

Ensuring binding with respect to maliciously generated keys Intuitively, this can be achieved by ensuring that no weak keys (that have behave differently than normal keys) exist, or are rejected by the decapsulation.

7 Conclusion

We introduced a new family of security notions for KEMs, that capture relevant binding properties.

Our binding properties can be used to prove the absence of attacks such as re-encapsulation attacks, or the absence of weak keys. KEM primitives that meet our binding properties are harder to misuse, at it seems that they can be met with minimal (and efficient) changes to existing KEMs.

We showed that for certain restricted classes of KEMs, such as implicitly-rejecting KEMs (which from our perspective resemble implicitly authenticated key exchanges), only four of our properties are relevant.

Our work is in line with a wider trend of constructing cryptographic primitives that are harder to misuse, and offer cleaner behavior with fewer side-cases. In particular, the guarantees offered by our properties perform a similar role as exclusive ownership and message-binding properties of digital signatures, and the various robustness notions defined for authenticated encryption schemes.

References

[1] Michel Abdalla, Mihir Bellare, and Gregory Neven. Robust Encryption. Cryptology ePrint Archive, Paper 2008/440, 2008. `https://eprint.iacr.org/2008/440`.

[2] Martin Albrecht, Carlos Cid, Kenneth G Paterson, Cen Jung Tjhai, and Martin Tomlinson. NTS-KEM. *NIST PQC Second Round*, 2, 2019. `https://nts-kem.io/` (Accessed December 2023).

[3] Nicolas Aragon, Paulo SLM Barreto, Slim Bettaieb, Loic Bidoux, Olivier Blazy, Jean-Christophe Deneuville, Philippe Gaborit, Shay Gueron, Tim Guneysu, Carlos Aguilar Melchor, et al. BIKE: bit flipping key encapsulation. 2017.

[4] Mihir Bellare, Hannah Davis, and Felix Günther. Separate Your Domains: NIST PQC KEMs, Oracle Cloning and Read-Only Indifferentiability. In Anne Canteaut and Yuval Ishai, editors, *Advances in Cryptology – EUROCRYPT 2020*, pages 3–32, Cham, 2020. Springer International Publishing.

[5] Daniel J Bernstein, Tung Chou, Tanja Lange, Ingo von Maurich, Rafael Misoczki, Ruben Niederhagen, Edoardo Persichetti, Christiane Peters, Peter Schwabe, Nicolas Sendrier, et al. Classic McEliece: conservative code-based cryptography. *NIST submissions*, 2017.

[6] Simon Blake-Wilson and Alfred Menezes. Unknown key-share attacks on the station-to-station (STS) protocol. In *Public Key Cryptography: Second International Workshop on Practice and Theory in Public Key Cryptography, PKC'99 Kamakura, Japan, March 1–3, 1999 Proceedings 2*, pages 154–170. Springer, 1999.

[7] Joppe 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 *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pages 1006–1018, 2016.

[8] Joppe Bos, Leo Ducas, Eike Kiltz, T Lepoint, Vadim Lyubashevsky, John M. Schanck, Peter Schwabe, Gregor Seiler, and Damien Stehle. CRYSTALS - Kyber: A CCA-Secure Module-Lattice-Based KEM. In *2018 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 353–367, 2018.

[9] Jacqueline Brendel, Cas Cremers, Dennis Jackson, and Mang Zhao. The provable security of ed25519: theory and practice. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 1659–1676. IEEE, 2021.

[10] Ronald Cramer and Victor Shoup. Design and Analysis of Practical Public-Key Encryption Schemes Secure against Adaptive Chosen Ciphertext Attack. Cryptology ePrint Archive, Paper 2001/108, 2001. `https://eprint.iacr.org/2001/108`.

[11] Cas Cremers, Alexander Dax, Charlie Jacomme, and Mang Zhao. Automated Analysis of Protocols that use Authenticated Encryption: How Subtle AEAD Differences can impact Protocol Security. Cryptology ePrint Archive, Paper 2023/1246, 2023. `https://eprint.iacr.org/2023/1246`.

[12] Cas Cremers, Samed Düzlü, Rune Fiedler, Marc Fischlin, and Christian Janson. BUFFing signature schemes beyond unforgeability and the case of post-quantum signatures. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 1696–1714. IEEE, 2021.

[13] Yevgeniy Dodis, Paul Grubbs, Thomas Ristenpart, and Joanne Woodage. Fast message franking: From invisible salamanders to encryptment. In *Advances in Cryptology–CRYPTO 2018: 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19–23, 2018, Proceedings, Part I 38*, pages 155–186. Springer, 2018.

[14] Jan-Pieter D'Anvers, Angshuman Karmakar, Sujoy Sinha Roy, and Frederik Vercauteren. Saber: Module-LWR based key exchange, CPA-secure encryption and CCA-secure KEM. In *Progress in Cryptology–AFRICACRYPT 2018: 10th International Conference on Cryptology in Africa, Marrakesh, Morocco, May 7–9, 2018, Proceedings 10*, pages 282–305. Springer, 2018.

[15] Eiichiro Fujisaki and Tatsuaki Okamoto. Secure integration of asymmetric and symmetric encryption schemes. In *Annual international cryptology conference*, pages 537–554. Springer, 1999.

[16] Paul Grubbs, Jiahui Lu, and Thomas Ristenpart. Message Franking via Committing Authenticated Encryption. Cryptology ePrint Archive, Paper 2017/664, 2017. `https://eprint.iacr.org/2017/664`.

[17] Paul Grubbs, Varun Maram, and Kenneth G. Paterson. Anonymous, Robust Post-Quantum Public Key Encryption. Cryptology ePrint Archive, Paper 2021/708, 2021. `https://eprint.iacr.org/2021/708`.

[18] Dennis Hofheinz, Kathrin Hövelmanns, and Eike Kiltz. A Modular Analysis of the Fujisaki-Okamoto Transformation. In Yael Kalai and Leonid Reyzin, editors, *Theory of Cryptography*, pages 341–371, Cham, 2017. Springer International Publishing.

[19] Andreas Hülsing, Joost Rijneveld, John Schanck, and Peter Schwabe. High-speed key encapsulation from NTRU. In *International Conference on Cryptographic Hardware and Embedded Systems*, pages 232–252. Springer, 2017.

[20] Dennis Jackson, Cas Cremers, Katriel Cohn-Gordon, and Ralf Sasse. Seems legit: Automated analysis of subtle attacks on protocols that use signatures. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pages 2165–2180, 2019.

[21] Julia Len, Paul Grubbs, and Thomas Ristenpart. Authenticated Encryption with Key Identification. Cryptology ePrint Archive, Paper 2022/1680, 2022. `https://eprint.iacr.org/2022/1680`.

[22] Carlos Aguilar Melchor, Nicolas Aragon, Slim Bettaieb, Loıc Bidoux, Olivier Blazy, Jean-Christophe Deneuville, Philippe Gaborit, Edoardo Persichetti, Gilles Zémor, and IC Bourges. Hamming quasi-cyclic (HQC). *NIST PQC Round*, 2(4):13, 2018.

[23] Payman Mohassel. A Closer Look at Anonymity and Robustness in Encryption Schemes. In *Advances in Cryptology - ASIACRYPT 2010 - 16th International Conference on the Theory and Application of Cryptology and Information Security*, volume 6477 of *Lecture Notes in Computer Science*, pages 501–518. Springer, 2010.

[24] NIST. NIST Post-Quantum Cryptography. `https://csrc.nist.gov/projects/post-quantum-cryptography`. Accessed: 2023-05-01.

[25] Thomas Pornin and Julien P Stern. Digital signatures do not guarantee exclusive ownership. In *Applied Cryptography and Network Security: Third International Conference, ACNS 2005, New York, NY, USA, June 7-10, 2005. Proceedings 3*, pages 138–150. Springer, 2005.

[26] Victor Shoup. A Proposal for an ISO Standard for Public Key Encryption. *IACR Cryptology ePrint Archive*, 2001:112, 2001.

A Definitions of our Binding Properties

In Figure 10, we show the honest versions of our binding properties. In Figure 11, we show the malicious versions of our binding properties.

$HON\text{-}BIND\text{-}K\text{-}CT^{\mathsf{KEM}}_{\mathcal{A}}$:

$\mathsf{sk}_0, \mathsf{pk}_0 \leftarrow \mathsf{KeyGen}()$
$\mathsf{sk}_1, \mathsf{pk}_1 \leftarrow \mathsf{KeyGen}()$
$\mathsf{ct}_0, \mathsf{ct}_1 \leftarrow \mathcal{A}^{D_b(\mathsf{sk}_b, \cdot)}(\mathsf{pk}_0, \mathsf{pk}_1)$
$\mathsf{k}_0 \leftarrow \mathsf{KEM.Decaps}(\mathsf{sk}_0, \mathsf{pk}_0, \mathsf{ct}_0)$
$\mathsf{k}_1 \leftarrow \mathsf{KEM.Decaps}(\mathsf{sk}_1, \mathsf{pk}_1, \mathsf{ct}_1)$
if $k_0 = \bot \vee k_1 = \bot$
  return 0
return $k_0 = k_1 \wedge ct_0 \neq ct_1$

$HON\text{-}BIND\text{-}K\text{-}PK^{\mathsf{KEM}}_{\mathcal{A}}$:

$\mathsf{sk}_0, \mathsf{pk}_0 \leftarrow \mathsf{KeyGen}()$
$\mathsf{sk}_1, \mathsf{pk}_1 \leftarrow \mathsf{KeyGen}()$
$\mathsf{ct}_0, \mathsf{ct}_1 \leftarrow \mathcal{A}^{D_b(\mathsf{sk}_b, \cdot)}(\mathsf{pk}_0, \mathsf{pk}_1)$
$\mathsf{k}_0 \leftarrow \mathsf{KEM.Decaps}(\mathsf{sk}_0, \mathsf{pk}_0, \mathsf{ct}_0)$
$\mathsf{k}_1 \leftarrow \mathsf{KEM.Decaps}(\mathsf{sk}_1, \mathsf{pk}_1, \mathsf{ct}_1)$
if $k_0 = \bot \vee k_1 = \bot$
  return 0
return $k_0 = k_1 \wedge pk_0 \neq pk_1$

$HON\text{-}BIND\text{-}CT\text{-}PK^{\mathsf{KEM}}_{\mathcal{A}}$:

$\mathsf{sk}_0, \mathsf{pk}_0 \leftarrow \mathsf{KeyGen}()$
$\mathsf{sk}_1, \mathsf{pk}_1 \leftarrow \mathsf{KeyGen}()$
$\mathsf{ct} \leftarrow \mathcal{A}^{D_b(\mathsf{sk}_b, \cdot)}(\mathsf{pk}_0, \mathsf{pk}_1)$
$\mathsf{k}_0 \leftarrow \mathsf{KEM.Decaps}(\mathsf{sk}_0, \mathsf{pk}_0, \mathsf{ct})$
$\mathsf{k}_1 \leftarrow \mathsf{KEM.Decaps}(\mathsf{sk}_1, \mathsf{pk}_1, \mathsf{ct})$
if $k_0 = \bot \vee k_1 = \bot$
  return 0
return $pk_0 \neq pk_1$

$HON\text{-}BIND\text{-}CT\text{-}K^{\mathsf{KEM}}_{\mathcal{A}}$:

$\mathsf{sk}_0, \mathsf{pk}_0 \leftarrow \mathsf{KeyGen}()$
$\mathsf{sk}_1, \mathsf{pk}_1 \leftarrow \mathsf{KeyGen}()$
$\mathsf{ct} \leftarrow \mathcal{A}^{D_b(\mathsf{sk}_b, \cdot)}(\mathsf{pk}_0, \mathsf{pk}_1)$
$\mathsf{k}_0 \leftarrow \mathsf{KEM.Decaps}(\mathsf{sk}_0, \mathsf{pk}_0, \mathsf{ct})$
$\mathsf{k}_1 \leftarrow \mathsf{KEM.Decaps}(\mathsf{sk}_1, \mathsf{pk}_1, \mathsf{ct})$
if $k_0 = \bot \vee k_1 = \bot$
  return 0
return $k_0 \neq k_1$

$HON\text{-}BIND\text{-}K, CT\text{-}PK^{\mathsf{KEM}}_{\mathcal{A}}$:

$\mathsf{sk}_0, \mathsf{pk}_0 \leftarrow \mathsf{KeyGen}()$
$\mathsf{sk}_1, \mathsf{pk}_1 \leftarrow \mathsf{KeyGen}()$
$\mathsf{ct} \leftarrow \mathcal{A}^{D_b(\mathsf{sk}_b, \cdot)}(\mathsf{pk}_0, \mathsf{pk}_1)$
$\mathsf{k}_0 \leftarrow \mathsf{KEM.Decaps}(\mathsf{sk}_0, \mathsf{pk}_0, \mathsf{ct})$
$\mathsf{k}_1 \leftarrow \mathsf{KEM.Decaps}(\mathsf{sk}_1, \mathsf{pk}_1, \mathsf{ct})$
if $k_0 = \bot \vee k_1 = \bot$
  return 0
return $k_0 = k_1 \wedge pk_0 \neq pk_1$

Figure 10: We argue that these five binding properties are desirable for a KEM. *HON-BIND-K-CT* corresponds to entry 1 from Table 1. *HON-BIND-K-PK* corresponds to entry 2 from Table 1. *HON-BIND-CT-K* corresponds to entry 3 from Table 1. *HON-BIND-CT-PK* corresponds to entry 4 from Table 1 and SROB from [17]. *HON-BIND-K, CT-PK* corresponds to entry 5 from Table 1 and SCFR from [17].

$MAL\text{-}BIND\text{-}K\text{-}CT_{\mathcal{A}}^{\mathsf{KEM}}$:

$(\mathsf{sk}_0, \mathsf{pk}_0), (\mathsf{sk}_1, \mathsf{pk}_1), \mathsf{ct}_0, \mathsf{ct}_1 \leftarrow \mathcal{A}()$
$\mathsf{k}_0 \leftarrow \mathsf{KEM.Decaps}(\mathsf{sk}_0, \mathsf{pk}_0, \mathsf{ct}_0)$
$\mathsf{k}_1 \leftarrow \mathsf{KEM.Decaps}(\mathsf{sk}_1, \mathsf{pk}_1, \mathsf{ct}_1)$
if $k_0 = \bot \vee k_1 = \bot$
  return 0
return $k_0 = k_1 \wedge ct_0 \neq ct_1$

$MAL\text{-}BIND\text{-}K\text{-}PK_{\mathcal{A}}^{\mathsf{KEM}}$:

$(\mathsf{sk}_0, \mathsf{pk}_0), (\mathsf{sk}_1, \mathsf{pk}_1), \mathsf{ct}_0, \mathsf{ct}_1 \leftarrow \mathcal{A}()$
$\mathsf{k}_0 \leftarrow \mathsf{KEM.Decaps}(\mathsf{sk}_0, \mathsf{pk}_0, \mathsf{ct}_0)$
$\mathsf{k}_1 \leftarrow \mathsf{KEM.Decaps}(\mathsf{sk}_1, \mathsf{pk}_1, \mathsf{ct}_1)$
if $k_0 = \bot \vee k_1 = \bot$
  return 0
return $k_0 = k_1 \wedge pk_0 \neq pk_1$

$MAL\text{-}BIND\text{-}CT\text{-}PK_{\mathcal{A}}^{\mathsf{KEM}}$:

$(\mathsf{sk}_0, \mathsf{pk}_0), (\mathsf{sk}_1, \mathsf{pk}_1), \mathsf{ct} \leftarrow \mathcal{A}()$
$\mathsf{k}_0 \leftarrow \mathsf{KEM.Decaps}(\mathsf{sk}_0, \mathsf{pk}_0, \mathsf{ct})$
$\mathsf{k}_1 \leftarrow \mathsf{KEM.Decaps}(\mathsf{sk}_1, \mathsf{pk}_1, \mathsf{ct})$
if $k_0 = \bot \vee k_1 = \bot$
  return 0
return $pk_0 \neq pk_1$

$MAL\text{-}BIND\text{-}CT\text{-}K_{\mathcal{A}}^{\mathsf{KEM}}$:

$(\mathsf{sk}_0, \mathsf{pk}_0), (\mathsf{sk}_1, \mathsf{pk}_1), \mathsf{ct} \leftarrow \mathcal{A}()$
$\mathsf{k}_0 \leftarrow \mathsf{KEM.Decaps}(\mathsf{sk}_0, \mathsf{pk}_0, \mathsf{ct})$
$\mathsf{k}_1 \leftarrow \mathsf{KEM.Decaps}(\mathsf{sk}_1, \mathsf{pk}_1, \mathsf{ct})$
if $k_0 = \bot \vee k_1 = \bot$
  return 0
return $k_0 \neq k_1$

$MAL\text{-}BIND\text{-}K, CT\text{-}PK_{\mathcal{A}}^{\mathsf{KEM}}$:

$(\mathsf{sk}_0, \mathsf{pk}_0), (\mathsf{sk}_1, \mathsf{pk}_1), \mathsf{ct} \leftarrow \mathcal{A}()$
$\mathsf{k}_0 \leftarrow \mathsf{KEM.Decaps}(\mathsf{sk}_0, \mathsf{pk}_0, \mathsf{ct})$
$\mathsf{k}_1 \leftarrow \mathsf{KEM.Decaps}(\mathsf{sk}_1, \mathsf{pk}_1, \mathsf{ct})$
if $k_0 = \bot \vee k_1 = \bot$
  return 0
return $k_0 = k_1 \wedge pk_0 \neq pk_1$

Figure 11: The malicious versions of the binding properties we present in Figure 10.

B Changelog

- Version 0.1, December 20, 2023: Initial draft.
- Version 0.1.1, December 22, 2023: Fixed typos, changed symbols for malicous and honest from M, and H to MAL, and HON, renamed Alex, Blake, and Charlie to A, B, and C. Added Changelog.