Security of Symmetric Primitives against Key-Correlated Attacks

Aisling Connolly\textsuperscript{1,2}, Pooya Farshim\textsuperscript{2,3}, and Georg Fuchsbauer\textsuperscript{2,3}

\textsuperscript{1} Ingenico Group, Paris, France
\textsuperscript{2} DI/ENS, CNRS, PSL University, Paris, France
\textsuperscript{3} Inria, Paris, France
\{aisling.connolly,pooya.farshim,georg.fuchsbauer\}@ens.fr

Abstract. We study the security of symmetric primitives against key-correlated attacks (KCA), whereby an adversary can arbitrarily correlate keys, messages, and ciphertexts. Security against KCA is required whenever a primitive should securely encrypt key-dependent data, even when it is used under related keys. KCA is a strengthening of the previously considered notions of related-key attack (RKA) and key-dependent message (KDM) security. This strengthening is strict, as we show that 2-round Even–Mansour fails to be KCA secure even though it is both RKA and KDM secure. We provide feasibility results in the ideal-cipher model for KCAs and show that 3-round Even–Mansour is KCA secure under key offsets in the random-permutation model. We also give a natural transformation that converts any authenticated encryption scheme to a KCA-secure one in the random-oracle model. Conceptually, our results allow for a unified treatment of RKA and KDM security in idealized models of computation.

Keywords: Key-correlated attack · related-key attack · key-dependent-message attack · ideal-cipher model · random-oracle model · authenticated encryption · xkcd.

1 Introduction

Cryptographic algorithms are subject to a multitude of threats. Many of these threats are accounted for in the theoretical security analysis carried out by cryptographers, but not all. For example, early on, the seminal paper of Goldwasser and Micali [GM84] pointed out that guarantees of semantic security may break down if the adversary sees encryptions of the secret key. Formal analyses of protocols can also become moot [Bih94b, Bih94a] when the assumption that cryptosystems are run on independently generated keys no longer holds. A number of works have analyzed the security of cryptosystems in the presence of key-dependent messages or when different keys are generated in dependent ways (see the related work section below). We continue this line of work and ask to what extent basic cryptosystems (such as blockciphers and symmetric encryption) can resist attacks that exploit correlated inputs.

1.1 Motivation

Our motivation for studying correlated-input security is twofold. We are interested in settings where a cryptosystem may be run on related keys—either by design or due to attacks—to securely encrypt messages that depend on the encryption key. Suppose a user stores a secret key $K$ on its hard drive. An adversary may be able to tamper with this key, for example flip some of its bits and change it to $K \oplus \Delta$ for some bit string $\Delta$. It may then obtain a full-disk encryption under this key. It is not clear what security assertions
can be made, as this setting falls outside both the related-key attack (RKA) and the key-dependent message (KDM) models. Indeed, the RKA model only allows the adversary to obtain encryptions of the form $\text{Enc}(\phi(K), M)$, for functions $\phi$ mapping keys to keys, but for key-independent messages, while KDM accounts for key-dependent encryptions of type $\text{Enc}(K, \psi(K))$, for functions $\psi$ mapping keys to messages, but under untampered keys. In the described attack, the adversary obtains $\text{Enc}(K \oplus \Delta, K \oplus \Delta)$. This is not covered by either of these models since both the key and the message are correlated with the original key. Other applications of KCAs include efficient garbling of XORs [App16], where KCA security (called RK-KDM there) with respect to linear functions or the form $\alpha \cdot K \oplus \Delta$ for a bit $\alpha$ are used.

These settings require a stronger security notion, which is what we introduce here. The Key-Correlated Attacks (KCAs) model lets the adversary obtain encryptions of key-dependent messages under related keys. Generally, wherever there is a possibility of both RKAs and KDM attacks, i.e., key-correlated encryptions of the form $\text{Enc}(\phi(K), \psi(K))$, there is good chance that the actual security needed is KCA security. A typical use case is when the round functions of a block cipher are keyed via related keys, and the construction is used to encrypt key-dependent data.

In our model, for generality, simplicity and strength, we symmetrically allow for key-dependent ciphertexts, that is, the adversary can see $\text{Dec}(\phi(K), \psi(K))$. Such settings arise when the decryption algorithm of a blockcipher is run during encryption, which is for example the case in the triple DES construction [BR06], the ElmD construction [BDMN16], or in amplification theorems for blockciphers [MP04, CPS14].

Our second motivation is conceptual in that KCA provides a unified approach to RKA and KDM security analyses of symmetric primitives. More concretely, our goal is to prove KCA feasibility theorems and then derive RKA and KDM security as simple corollaries. This allows for reuse of security proofs and identifies classes of permitted attacks more generally, while leading to stronger security results.

1.2 Related work

RKA security. Knudsen and Biham [Knu93, Bih94b, Bih94a] initiated the study of RKAs and Bellare and Kohno [BK03] gave a theoretical treatment. High-profile RKAs on AES were discovered by Biryukov et al. [BKN09, BK09]. The RKA model was extended by Albrecht et al. [AFPW11] to account for attacks that depend on an ideal primitive [Har09, Ber10]. The RKA security of Feistel networks [BF15] and Even–Mansour ciphers [FP15, CS15] have been studied. Bellare, Cash, and Miller [BCM11] present a comprehensive treatment of RKA security for various cryptographic primitives.


Correlated inputs. Study of security under correlated inputs goes back to the work of Ishai et al. [IKNP03] as correlation-robustness in the design of oblivious transfer protocols. Correlated-input security was made explicit for hash functions by Goyal, O’Neill, and
Rao [GOR11], who show relations with related-key attacks. The work of Böhl, Davies and Hofheinz [BDH14] considers related-key attacks in the presence of key-dependent messages. Their RKA-KDM security could be considered a natural analogue of our model for public-key encryption. They construct schemes that achieve their notion based on number-theoretic assumptions such as DDH, LWE, QR, or DCR. Applebaum [App16] gives an RKA-KDM symmetric encryption scheme based on the LPN assumption.

1.3 Contributions

Building on the above line of works, we formulate a new security model incorporating and strengthening both the RKA and KDM models. We speak of key-correlated attack (KCA) in this context, a name that is loosely inspired by the introduction of correlated-input attacks against hash functions [GOR11] (note that the notion of key-dependent input attacks has already been used by Halevi and Krawczyk [HK07]). We give appropriate definitions of security under key-correlated attacks that relate well to the standard RKA and KDM security notions. Our definition extends that in [App16] for randomized symmetric encryption under chosen-plaintext attacks to the setting of authenticated encryption with associated data (AEAD).

We start with comparing our notion to existing ones. After proving that KCA implies RKA and KDM security, we show that KCA security is strictly stronger than even simultaneously having RKA and KDM security. We give a natural separation by demonstrating a KCA attack on the 2-round Even–Mansour cipher, which was shown to satisfy both RKA and KDM security in two previous works [FP15, FKV17].

After defining KCA and showing a separation result, we study feasibility of KCAs. Our starting point is the ideal-cipher model, in which all parties have oracle access to a keyed random permutation in both directions. We cannot allow arbitrary dependencies of keys and messages as otherwise “trivial” attacks, which work against any scheme, arise. To exclude these and thus obtain a meaningful notion, we restrict the classes of allowed dependencies. We show that if they satisfy appropriate notions of key-unpredictability and claw-freeness then the ideal cipher satisfies KCA security. Roughly speaking, key-unpredictability requires that the adversary does not obtain encryptions or decryption under predictable keys. Claw-freeness, on the other hand, requires that the inputs are distinct, and so repetition pattern of the outputs cannot be exploited. Analogues of these notions were previously considered in the RKA and KDM settings.

In our setting we require a third condition, which we call cross-key-claw detectability, that allows us to deal with claws across encryption and decryption queries. This notion is sufficiently weak so that in the RKA and KDM setting it automatically follows from claw-freeness. In the KCA setting, however, it does not necessarily, as it restricts claws on keys.\(^1\) Two results on the RKA and KDM security of the ideal cipher by Bellare and Kohno [BK03] and Farshim, Khati, Vergnaud [FKV17] respectively, fall out as natural corollaries of our theorem.

Turning to concrete constructions, we analyze the KCA security of the iterated Even–Mansour cipher with three rounds in the random-permutation model. We show that with reuse of keys (which is known to be necessary for RKA security [FP15]) and using different permutations (which is necessary for KDM security whenever keys are reused [BW99, FKV17]) we can provably achieve security against key-correlated attacks that concurrently encrypt messages \(M\) or offsets of key of the form \(K \oplus \Delta_2\) under other offsets of the key of the form \(K \oplus \Delta_1\). This strengthens two feasibility results due to Farshim, Khati, Vergnaud [FKV17] and Farshim and Procter [FP15].

\(^1\)Claw-freeness can be modified to key-claw-freeness across encryption and decryption so that cross-key-claw-freeness is automatic. But in this case the reach of our feasibility results do not extend to the KDM setting since under KDM keys always collide.
From a technical point of view, the novelty of our KCA proof for 3-round Even–Mansour is that we keep the outer permutations partially consistent with the replaced forgetful oracles as well as the permutation oracles. For legal queries we show this can be done with overwhelming probability, while a detection algorithm will allow us to identify illegal queries and reject them. This proof thus deviates from previous works in which oracles are fully decoupled. As a result we also obtain a different (albeit somewhat more complex) way to prove the RKA security of 3-round Even–Mansour against key offsets by replacing the outer (rather the inner) permutation [FP15].

We end the paper by showing how to generically transform any AE-secure AEAD scheme to one which is a KCA-secure in the random-oracle model by hashing the key with nonces. For this result we only require the set of allowed functions to be unpredictable, as nonces automatically prevent repetitions due to claws in the functions. In contrast to previous work by Bellare and Keelveedhi [BK11] on similar transforms for achieving KDM-security, our scheme is secure with key-dependent nonces and headers. Although key-dependent headers are briefly discussed in [BK11], security with respect to key-dependent nonces is not considered at all. Arguably, however, there is a stronger case for the key-dependency of nonces than of headers: when nonces are randomly chosen they might become correlated with the key due to, e.g., bad generation of random numbers. For key-dependent headers, Bellare and Keelveedhi give a negative result, by having the adversary exploit the pattern of decryption errors (either ⊥ for an illegal query or 0 for failure in authenticity) to recover the key. In our setting, however, the decryption oracle only returns a single error symbol, which enables security under key-dependent inputs. If our model were modified to also have distinct error symbols, an attack similar to that in [BK11] would arise. We note that in these settings one might be able to obtain non-trivial feasibility results by requiring a form claw-freeness.2

Our work leaves a number of open questions to be addressed in this area: How many rounds are needed for other ciphers (such as Feistel networks) in order to achieve provable KCA security? Do modes of operation (such as CBC) provide KCA security assuming that their underlying blockciphers are KCA secure? Does KCA help in the cryptanalysis of concrete blockciphers?

**Organization.** We start with basic notation and preliminaries in Sec. 2. In Sec. 3, we define KCA security for blockciphers and study its relation to RKA and KDM security before showing two separation results in Sec. 4. In Sec. 5 we study KCA in the ideal-cipher model and in Sec. 6 we prove 3-round Even–Mansour KC-CCA secure for offsets. Sec. 7 contains our generic construction of a KCA-secure authenticated encryption scheme from any AE-secure one.

## 2 Preliminaries

**Notation.** We let \{0, 1\}^n denote the set of bit strings of length n and \{0, 1\}^* the set of all finite-length bit strings. For two bit strings X and Y, X \| Y denotes their concatenation and (X, Y) denotes a uniquely (and efficiently) decodable encoding of X and Y. The length of X is denoted by |X|. By x \leftarrow S we mean sampling x uniformly from set S, whereas by y \leftarrow A(x) we mean the action of assigning the output of the randomized algorithm A on input x to y. We denote appending element X (resp., a list L') to a list L by L : X (resp., L : L').

We adopt the code-based game-playing language of Bellare and Rogaway [BR06], for which all lists are initialized to empty and all bad flags to false. Following the concrete security approach, we will primarily content ourselves with defining advantages and providing concrete reductions, without dwelling too much on the question when a

2Consistently, the attack in [BK11] exploits claws in the key-dependent headers.
scheme is actually deemed secure or not (e.g., for a sufficiently large class of adversaries, the advantages are sufficiently small). One can of course easily recast our work in an asymptotic framework (where for all probabilistic polynomial-time adversaries advantages should be negligible in the security parameter $\lambda$, i.e., in $\lambda^{-\omega(1)}$).

**Blockciphers**. Given a non-empty finite set $\mathcal{K}$ and a non-empty set $\mathcal{M}$, called the key space and the message space respectively, we let $\text{Block}(\mathcal{K}, \mathcal{M})$ denote the set of all functions $E : \mathcal{K} \times \mathcal{M} \to \mathcal{M}$ such that for each $K \in \mathcal{K}$ the map $E(K, \cdot)$ is (1) a permutation on $\mathcal{M}$ and (2) length-preserving in the sense that for all $M \in \mathcal{M}$ we have that $|E(K, M)| = |M|$. Such an $E$ uniquely defines its inverse $D : \mathcal{K} \times \mathcal{M} \to \mathcal{M}$. A blockcipher for key space $\mathcal{K}$ and message space $\mathcal{M}$ is a tuple of efficient algorithms $\mathcal{BC} := (E, D)$ such that $E \in \text{Block}(\mathcal{K}, \mathcal{M})$ and $D$ is its inverse. We assume that the keys of a blockcipher are chosen uniformly from the key space $\mathcal{K}$, which is typically $\{0, 1\}^k$ for some $k \in \mathbb{N}$ called the key length. Algorithm $E$ is the deterministic encryption algorithm $E : \mathcal{K} \times \mathcal{M} \to \mathcal{M}$. Typically $\mathcal{M} = \{0, 1\}^n$ for some $n \in \mathbb{N}$ called the block length. Algorithm $D$ is the deterministic decryption algorithm $D : \mathcal{K} \times \mathcal{M} \to \mathcal{M}$. A blockcipher is correct if for all $K \in \mathcal{K}$ and $M \in \mathcal{M}$ we have $D(E(K, M)) = M$ and also $E(K, \cdot)$ is length-preserving (which follows from decryptability if $\mathcal{M} = \{0, 1\}^n$). A (public) permutation on $\mathcal{M}$ is a blockcipher with a singleton key space $\mathcal{K} = \{\varepsilon\}$. For variable input length blockciphers, we require that for each $n$, $\{0, 1\}^n$ is either contained in $\mathcal{M}$, or $\{0, 1\}^n \cap \mathcal{M} = \emptyset$. We denote a permutation with $P$ and its inverse with $P^{-1}$.

**Ideal ciphers**. The ideal cipher for key space $\mathcal{K}$ and message space $\mathcal{M}$ is the uniform distribution over $\text{Block}(\mathcal{K}, \mathcal{M})$. The ideal-cipher model (ICM) for given key and message spaces $\mathcal{K}, \mathcal{M}$ is a model of computation where all parties, honest or otherwise, have oracle access to a uniformly random element in $\text{Block}(\mathcal{K}, \mathcal{M})$ and its inverse. The ideal-cipher model when restricted to $\mathcal{K} = \{\varepsilon\}$ gives rise to the random-permutation model (RPM). We abbreviate $\text{Block}(\{0, 1\}^k, \{0, 1\}^n)$ by $\text{Block}(k, n)$ and $\text{Block}(\{\varepsilon\}, \{0, 1\}^n)$ by $\text{Perm}(n)$.

### 3 The Model

#### 3.1 Concept and definitions

**Correlation-derivation function (CDF)**. A correlation-derivation function (CDF) is a circuit of the form

$$\xi : \mathcal{K} \longrightarrow \mathcal{K} \times \mathcal{M}.$$  

A set of such functions is called a CDF set. Throughout the paper we denote CDF sets by $\Xi$ and require that membership of a CDF set can be efficiently decided. We will define key-correlated security primitives relative to two CDF sets $\Xi^e$ and $\Xi^d$ that describe allowed encryption and decryption queries, respectively.

**KCA-secure blockciphers**. Let $\mathcal{BC}$ be a blockcipher with key space $\mathcal{K} = \{0, 1\}^k$ and message (and ciphertext) space $\mathcal{M} = \{0, 1\}^n$. Let $\Xi^e$ and $\Xi^d$ be CDF sets for the input space $\mathcal{K} \times \mathcal{M}$. The KC-CCA advantage of an adversary $\mathcal{A}$ against $\mathcal{BC}$ is defined by

$$\text{Adv}^{\text{KCCA}}_{\mathcal{BC}}(\Xi^e, \Xi^d, A) := 2 \cdot \Pr \left[ \text{KC-CCA}_{\mathcal{BC}}^{\Xi^e, \Xi^d, A} = 1 \right],$$  

where game $\text{KC-CCA}_{\mathcal{BC}}^{\Xi^e, \Xi^d, A}$ is given in Fig. 1. In this game, the adversary’s goal is to decide whether (case $b = 1$) its oracles are using the blockcipher $\mathcal{BC} = (E, D)$ or (case $b = 0$, the ideal case) a random permutation $iE$ and its inverse $iD$. Lists $\text{ML}$ and $\text{CL}$ prevent the adversary from trivially winning the game. Otherwise the adversary could, for instance, recover the challenge key $K^*$ by querying an encryption (or decryption) of $K^*$ and subsequently ask for decryption (or encryption) of the oracle’s reply.
As examples of KC functions, suppose that related keys, described by functions from some set $\Phi$, are used within the specification of an encryption scheme (an example are the 3GPP protocols [IK04]). Suppose further that the scheme is used to encrypt messages that depend on the key in ways represented by functions from a set $\Psi$. The overall effect is that an adversary has access to ciphertexts corresponding to key-dependent messages under related keys. In other words, the adversary can see key-correlated ciphertexts for functions:

\[ \xi : K \mapsto (\phi(K)) \text{ with } \phi \in \Phi \text{ and } \psi \in \Psi, \quad \text{and} \]
\[ \xi : K \mapsto (\phi(K), M) \text{ with } \phi \in \Phi \text{ and } M \in M. \]

Alternatively, suppose an adversary has access to encryptions under related keys with respect to $\Phi$ through, say, injection of faults that change bits of a hardware-stored encryption key [BDL07, BS97]. If the scheme is used to encrypt key-dependent messages with respect to $\Psi$, the adversary would be able to launch a KCA for the functions:

\[ \xi : K \mapsto (\phi(K), \psi(\phi(K))) \text{ with } \phi \in \Phi \text{ and } \psi \in \Psi, \quad \text{and} \]
\[ \xi : K \mapsto (\phi(K), M) \text{ with } \phi \in \Phi \text{ and } M \in M. \]

The KCA model thus captures, among other things, a variety of joint RKA and KDM attacks on a blockcipher.

### 3.3 Relation with RKA and KDM

Let $id$ denote the identity function on a key space $K$ and let $\Gamma$ denote the set of all constant functions $K \mapsto M$ for $M \in M$. We identify a pair of functions $(\phi, \psi)$, with ranges $K$ and $M$ respectively, with the correlation-derivation function $K \mapsto (\phi(K), \psi(K))$.

**CPA/CCA-only sets.** We call a pair $(\Xi^e, \Xi^d)$ CPA-only if $\Xi^e = \Xi^d = \{id\} \times \Gamma$. (The adversary can only make regular encryption and decryption queries.) We call $(\Xi^e, \Xi^d)$ CPA-only if $\Xi^e = \{id\} \times \Gamma$ and $\Xi^d = \emptyset$.

![Game defining the $(\Xi^e, \Xi^d)$-KC-CCA security of a blockcipher BC = (E, D) with key space K and message space M. We require that $\xi^e \in \Xi^e$ and $\xi^d \in \Xi^d$ for all queries.](Image)

We note that for two sets $\Xi^e_1 \subseteq \Xi^e_2$ and $\Xi^d_1 \subseteq \Xi^d_2$ security against $(\Xi^e_2, \Xi^d_2)$-KC-CCA implies security against $(\Xi^e_1, \Xi^d_1)$-KC-CCA. Whenever $\Xi^d = \emptyset$ we obtain a chosen-plaintext attack model.

**Note.** Analogues of KCA security can be formulated for hash functions and pseudorandom generators, which become equivalent to correlated-input security for hash functions [GOR11] and RKA security for PRGs [BCM11].

#### 3.2 Examples

As examples of KC functions, suppose that related keys, described by functions from some set $\Phi$, are used within the specification of an encryption scheme (an example are the 3GPP protocols [IK04]). Suppose further that the scheme is used to encrypt messages that depend on the key in ways represented by functions from a set $\Psi$. The overall effect is that an adversary has access to ciphertexts corresponding to key-dependent messages under related keys. In other words, the adversary can see key-correlated ciphertexts for functions:

\[ b \leftarrow \{0,1\} \]
\[ (iE, iD) \leftarrow \text{Block}(K, M) \]
\[ K \leftarrow K \]
\[ b' \leftarrow A^{\text{KENC, KDEC}} \]
\[ \text{return } (b' = b) \]

We identify a pair of functions $\xi^e$ and $\xi^d$ respectively, with the correlation-derivation function $\xi^d \mapsto K$.

<table>
<thead>
<tr>
<th>Game $\text{KC-CCA}^{\Xi^e, \Xi^d, A}$:</th>
<th>Proc. $\text{KDEC}(\xi^d)$:</th>
<th>Proc. $\text{KENC}(\xi^e)$:</th>
</tr>
</thead>
<tbody>
<tr>
<td>$b \leftarrow {0,1}$</td>
<td>$(k, C) \leftarrow \xi^d(K)$</td>
<td>$(K, M) \leftarrow (\phi(K))$</td>
</tr>
<tr>
<td>$(iE, iD) \leftarrow \text{Block}(K, M)$</td>
<td>if $(k, C) \in \emptyset$: return $\perp$</td>
<td>if $(K, M) \in \emptyset$: return $\perp$</td>
</tr>
<tr>
<td>$K \leftarrow K$</td>
<td>$C \leftarrow E(K, M)$</td>
<td>$M \leftarrow D(K, M)$</td>
</tr>
<tr>
<td>$b' \leftarrow A^{\text{KENC, KDEC}}$</td>
<td>if $b = 0$: $C \leftarrow iE(K, M)$</td>
<td>if $b = 0$: $M \leftarrow iD(K, M)$</td>
</tr>
<tr>
<td>\text{return } ($b' = b$)</td>
<td>\text{CL} $\leftarrow \text{CL}$ $(K, C)$</td>
<td>\text{ML} $\leftarrow \text{ML}$ $(K, M)$</td>
</tr>
</tbody>
</table>

$\xi^e \in \Xi^e$ and $\xi^d \in \Xi^d$ for all queries.
RKA/KDM-only sets. We call a pair \((\Xi^e, \Xi^d)\) RKA-only if \(\Xi^e = \Xi^d = \Phi \times \Gamma\) for an RKA set \(\Phi\) of functions mapping keys to keys. We call \((\Xi^e, \Xi^d)\) KDM-only if \(\Xi^e = \{\text{id}\} \times \Psi\) and \(\Xi^d = \{\text{id}\} \times \Gamma\) (i.e., no key-dependent ciphertexts allowed [FKV17]) for a KDM set \(\Psi\) of functions mapping keys to messages. (The CPA versions are defined analogously by demanding that \(\Xi^d = \emptyset\).) We assume that any KDM set \(\Psi\) contains \(\Gamma\), as it is natural to always allow for chosen plaintexts.

We show that our definition of KC security extends the standard RKA and KDM definitions for blockciphers, which we recall in Appendix A. In particular, for KDM-only sets we have that KCA and KDM security are equivalent and similarly for RKA-only sets KCA and RKA security are equivalent. For two of the four implications, we assume claw-freeness of the correlation-derivation functions. This requires that it is hard to find two different functions that have the same output given a random input and is defined formally in Sec. 5 (p. 10). Claw-freeness is required because the KCA game returns ⊥ whenever claws are detected whereas the RKA game does not. The proof of the following proposition can be found in Appendix A.

**Proposition 1.** Let BC be a blockcipher. Let \(\Xi^e\) and \(\Xi^d\) be two CDF sets.

**RKA:** Suppose \((\Xi^e, \Xi^d)\) are RKA-only with \(\Xi^e = \Xi^d = \Phi \times \Gamma\). If BC is \((\Xi^e, \Xi^d)\)-KC-CCA-secure then it is \(\Phi\)-RK-CCA-secure. If BC is \(\Phi\)-RK-CCA-secure and \(\Xi^e = \Xi^d\) is claw-free then BC is \((\Xi^e, \Xi^d)\)-KC-CCA-secure.

In particular, let \(n\) be the block length, \(A\) be an adversary and \(q\) the maximum number of its queries. Then there exists an adversary \(B\) such that

\[
\text{Adv}_{\text{RK-CCA}}(\Phi, A) \leq \text{Adv}_{\text{KC-CCA}}(\Xi^e, \Xi^d, B) + q^2/2^n.
\]

Moreover, there exist adversaries \(B\) and \(B^{cf}\) such that

\[
\text{Adv}_{\text{KC-CCA}}(\Xi^e, \Xi^d, A) \leq q^2 \cdot (\text{Adv}_{\Phi}^{cf}(B^{cf}) + 1/2^n) + \text{Adv}_{\text{RK-CCA}}(\Phi, B).
\]

**KDM:** Suppose \((\Xi^e, \Xi^d)\) are KDM-only with \(\Xi^e = \{\text{id}\} \times \Psi\) and \(\Xi^d = \{\text{id}\} \times \Gamma\). If BC is \(\Psi\)-KDM-CCA-secure then it is \((\Xi^e, \Xi^d)\)-KC-CCA-secure. If BC is \((\Xi^e, \Xi^d)\)-KC-CCA-secure and \(\Xi^e\) is claw-free then BC is \(\Psi\)-KDM-CCA-secure.

In particular, for every adversary \(A\) there exists an adversary \(B\) such that

\[
\text{Adv}_{\text{KC-CCA}}(\Xi^e, \Xi^d, A) = \text{Adv}_{\text{KDM-CCA}}(\Psi, B),
\]

and there exist adversaries \(B\) and \(B^{cf}\) such that

\[
\text{Adv}_{\text{KDM-CCA}}(\Psi, A) \leq q^2 \cdot \text{Adv}_{\Psi}^{cf}(B^{cf}) + \text{Adv}_{\text{KC-CCA}}(\Xi^e, \Xi^d, B),
\]

where \(q\) is the maximum number of queries adversary \(A\) makes.

Analogous implications hold in the CPA setting.

## 4 Key-Correlated Attacks

In this section we show that even if a blockcipher simultaneously achieves security against RK and KDM attacks, it may still fail to achieve security against concurrent RK/KDM attacks, and hence also fail to achieve KCA security. We provide two separations, one artificial and one natural, to demonstrate this.
The EM r-round iterated Even–Mansour cipher 

4.1 A generic separation result

Let $\mathcal{K} = \mathcal{M} = \{0,1\}^n$ and define the following sets of functions:

\[
\Phi^\oplus := \{ K \mapsto K \oplus M : M \in \mathcal{M} \}
\]

\[
\Psi^\oplus := \{ K \mapsto \alpha \cdot K \oplus M : M \in \mathcal{M}, \alpha \in \{0,1\} \}
\]

\[
\Xi^\oplus := \{ K \mapsto (K \oplus M_1, \alpha \cdot K \oplus M_2) : M_1, M_2 \in \mathcal{M}, \alpha \in \{0,1\} \} .
\]

Given a $\Phi^\oplus$-RKA secure and $\Psi^\oplus$-KDM secure blockcipher $(E,D)$, consider the modified cipher $(E',D')$ shown in Fig. 2, where $\min$ is the lexicographic minimum and $\overline{K} := K \oplus 1^k$. Note that $(E',D')$ is again a blockcipher, i.e., a permutation for each value of the key (we simply swapped $E(K,K)$ and $C^*$ in the image space).

To see that $(E',D')$ is not $(\Xi^\oplus,\emptyset)$-KC-CCA secure, consider an adversary that queries $K \mapsto (K,K)$ and $K \mapsto (\overline{K},\overline{K}) = (K \oplus 1^n, K \oplus 1^n)$ to its KCEnc oracle. For the modified cipher both queries yield the same result:

\[
E'(\overline{K},\overline{K}) = E(\min(\overline{K},\overline{K}),0^n) = E(\min(K,K),0^n) = E(K,K) ,
\]

while for the ideal cipher this would only happen with probability $1/2^n$. On the other hand, the modified cipher $(E',D')$ remains both RKA and KDM secure as it essentially behaves like $(E,D)$ when no joint RKA and KDM attacks as above can be mounted. We kept this discussion informal as our second separating example below is more natural and practically relevant.

4.2 Attack on 2-round Even–Mansour

The $r$-round Even–Mansour cipher [EM97, DR01, BKL\textsuperscript{+}12] is defined by $r$ permutations $P_1, \ldots, P_r$, has key space $\mathcal{K} = \{0,1\}^{(r+1)n}$, domain $\mathcal{M} = \{0,1\}^n$ and encryption and decryption algorithms (cf. Fig. 3)

\[
E((K_1, \ldots, K_{r+1}), M) := P_r(\ldots P_2(P_1(M \oplus K_1) \oplus K_2) \ldots) \oplus K_{r+1} ,
\]

\[
D((K_1, \ldots, K_{r+1}), C) := P_1(\ldots P_{r-1}(P_r(C \oplus K_{r+1}) \oplus K_r) \ldots) \oplus K_1 .
\]

The EM ciphers can also be considered in configurations where (some of the) keys and/or (some of the) permutations are reused in different rounds. We denote the EM cipher where $P_i$ and $K_{i+1}$ are used in round $i$ by $E_2^{P_1, \ldots, P_r}[K_1, K_2, \ldots, K_{r+1}]$. 

Figure 2: Blockcipher $(E',D')$ that is both KDM and RKA secure but not KCA secure.

Note that $\min(K,\overline{K}) = \min(\overline{K},\overline{K})$, thus $E'(K,K) = E'(\overline{K},\overline{K})$.

Figure 3: The $r$-round iterated Even–Mansour cipher.
Thus the two key-correlated ciphertexts with key reuse and independent permutations set Unpredictability. Those of both Bellare and Kohno \([BK03]\) and of Farshim et al. \([FKV17]\). We define four by checking whether the answers differ by a known value thus merely needs to make two queries. Ciphertexts would be randomly and independently distributed among the ideal cipher this event only occurs with probability \(\Phi\). We require that \(\Phi\) be RK-CPA secure in \([FP15]\) and \(\Phi\)-KDM-CCA secure in \([FKV17]\). We now show that this construction fails to be \((\Xi^e, \Xi^d)\)-KCCA secure. Consider the KDM encryption

\[
E(K, M) = P_2(P_1(0^n) \oplus K) \oplus K .
\]

Now let \(\Delta^* := P_1(0^n) \oplus P_1(1^n)\) and consider the key-correlated encryption

\[
C_1 := E(K \oplus \Delta^*, K \oplus \Delta^* \oplus 1^n) = P_2(P_1(1^n) \oplus K \oplus \Delta^*) \oplus K \oplus \Delta^*
\]

\[= P_2(P_1(0^n) \oplus K) \oplus K \oplus \Delta^*\]

\[\overset{(3)}{=} E(K, K) \oplus \Delta^* .\]

Thus the two key-correlated ciphertexts \(C_1\) and \(E(K, K)\) differ by a known value \(\Delta^*\). For the ideal cipher this event only occurs with probability \(1/2^n\) since \(\Delta^* \neq 0^n\) and the two ciphertexts would be randomly and independently distributed among \(2^n\) values. An attacker thus merely needs to make two queries \(K \mapsto (K, K)\) and \(K \mapsto (K \oplus \Delta^*, K \oplus \Delta^* \oplus 1^n)\) and check whether the answers differ by \(\Delta^*\). The advantage of this attacker is \(1 - 1/2^n\).

## 5 KCA Security of the Ideal Cipher

In this section we study the feasibility of achieving security against key-correlated attacks in the ideal-cipher model. In this setting the adversary has oracle access to the ideal cipher in both the forward and backward direction.

### 5.1 KCA Security of the Ideal Cipher

**KCA Security of the Ideal Cipher.** We begin by extending the standard notion of CCA security to the KCA setting, where key-dependent messages may be enciphered under related keys. Consider game \(\text{KC-CCA}^\Xi_{\text{Block}(k, n)}\) shown in Fig. 4. The KCA advantage of an adversary \(A\) against the ideal cipher with key length \(k\) and block length \(n\) is defined by

\[
\text{Adv}_{\text{Block}(k, n)}^{\text{KCA}}(\Xi^e, \Xi^d, A) := 2 \cdot \text{Pr}[\text{KC-CCA}^\Xi_{\text{Block}(k, n)}] - 1 .
\]

We now prove a feasibility theorem for the ideal cipher that generalizes and extends those of both Bellare and Kohno \([BK03]\) and of Farshim et al. \([FKV17]\). We define four conditions on CDF sets, which exclude trivial attacks.

**Unpredictability.** The unpredictability advantage of an adversary \(A\) against a CDF set \(\Xi\) is defined as

\[
\text{Adv}_{\Xi}^{\text{up}}(A) := \text{Pr} \left[ \xi(K) = (K_0, M_0) : (\xi, (K_0, M_0)) \leftrightarrow A; K \leftrightarrow K \right] ,
\]
where we require that $\xi \in \Xi$. Informally, we say $\Xi$ is unpredictable if the above advantage is “small” for every “reasonable” $A$.

We also define a multi-shot version of this game where $A$ outputs a list $L_1$ of candidates for $\xi$ and a list $L_2$ of predicted values $(K_0, M_0)$. Adversary $A$ wins if some $\xi$ in $L_1$ evaluates to some $(K_0, M_0)$ in $L_2$ when run on a random key $K$. Denoting the advantage in this game by $\text{Adv}^{\text{m-up}}_{\Xi}(A)$, a simple guessing argument shows that for any $A$ there exists $B$ with

$$\text{Adv}^{\text{m-up}}_{\Xi}(A) \leq \ell_1 \cdot \ell_2 \cdot \text{Adv}^{\text{up}}_{\Xi}(B),$$

where $\ell_1$ and $\ell_2$ are upper bounds on the sizes of $L_1$ and $L_2$ respectively.

**Key-unpredictability.** The key-unpredictability advantage of an adversary $A$ against a CDF set $\Xi$ is defined as

$$\text{Adv}^{\text{kup}}_{\Xi}(A) := \Pr[\xi_1(K) = K_0 : (\xi, K_0) \leftrightarrow A; K \leftrightarrow K],$$

where we require that $\xi \in \Xi$. Here $\xi_1(K)$ denotes the projection to the first coordinate of $\xi(K)$. Informally, we say $\Xi$ is key-unpredictable if the above advantage is “small” for every “reasonable” $A$.

We also define a multi-shot version of this game where $A$ outputs a list $L_1$ of candidates $\xi$ and a list $L_2$ of predicted values $K_0$. Adversary $A$ wins if for a random key $K$, for some $\xi$ in $L_1$ we have $\xi_1(K) = K_0$ for some $K_0$ in $L_2$. If we denote the advantage in this game by $\text{Adv}^{\text{m-kup}}_{\Xi}(A)$, again we have that there exists a single-shot $B$ such that

$$\text{Adv}^{\text{m-kup}}_{\Xi}(A) \leq \ell_1 \cdot \ell_2 \cdot \text{Adv}^{\text{kup}}_{\Xi}(B),$$

where $\ell_1$ and $\ell_2$ are upper bounds on the sizes of $L_1$ and $L_2$ respectively. Note that key-unpredictability implies unpredictability. The converse does not hold as $\xi : K \mapsto (0, K)$ is unpredictable but not key-unpredictable.

**Claw-freeness.** This requires that it is hard to find two distinct functions that output the same on a random input. Formally, the claw-free advantage of an adversary $A$ against a CDF set $\Xi$ is defined as

$$\text{Adv}^{\text{cf}}_{\Xi}(A) := \Pr[\xi \neq \xi' \land \xi(K) = \xi'(K) : (\xi, \xi') \leftrightarrow A; K \leftrightarrow K],$$

where we require that $\xi, \xi' \in \Xi$. We say $\Xi$ is claw-free if the above advantage is “small” for every “reasonable” $A$. Once again, a multi-shot version of the game with lists of candidates $L_1$ and $L_2$ for $\xi$ and $\xi'$ can be defined, and as before there exists $B$ with

$$\text{Adv}^{\text{m-cf}}_{\Xi}(A) \leq \ell_1 \cdot \ell_2 \cdot \text{Adv}^{\text{cf}}_{\Xi}(B).$$

**Cross-key-claw-detectability.** We introduce a new notion and call a pair of CDF sets $(\Xi^c, \Xi^d)$ cross-key-claw-detectable (xkcd)$^5$ if there is an efficient detection algorithm $\text{Det}$ such that for any $A$ the advantage below is small.

$$\text{Adv}^{\text{xkcd}}_{\Xi^c, \Xi^d, \text{Det}}(A) := \Pr[\text{Det}(\xi^c, C, \xi^d, M, 1) \neq [(\xi^c)_1(K), C] = \xi^d(K)] \vee (\xi^c, C, \xi^d, M) \leftrightarrow A; \quad K \leftrightarrow K].$$

We require that $\xi^c \in \Xi^c$ and $\xi^d \in \Xi^d$. Roughly speaking, xkcd provides a way to treat legality of queries across encryption and decryption queries in the KCA game. For

$^4$Single-shot adversary $B$ runs any given multi-shot $A$ to obtain two lists, picks one entry at random from each list and outputs the pair. Then the success probability of $B$ is at least $1/(\ell_1 \ell_2)$ times that of $A$.

$^5$Please do not confuse this notion with the (popular :) web-comic xkcd.com.
example, in an illegal decryption query the output of an earlier encryption query (stored by the game in list \( CL \), cf. Fig. 1) is sent to the decryption oracle through a CDF function whose key component matches that used in encryption. \( Det \) allows us to decide when such cases occur and deal with them appropriately in the security analysis.

A generalization of this notion considers lists \( L_1 \) for pairs \((\xi^e, C)\) and \( L_2 \) for \((\xi^d, M)\) and \( A \) wins if \( Det \) fails for any given tuple \((\xi^e, C, \xi^d, M)\) in the product of the lists. Again, a guessing argument shows that for every \( A \) there exists \( B \) with

\[
\text{Adv}^{\text{m-xkcd}}_{\Xi^e, \Xi^d, \text{Det}}(A) \leq \ell_1 \cdot \ell_2 \cdot \text{Adv}^{\text{kcd}}_{\Xi^e, \Xi^d, \text{Det}}(B).
\]

**Remark.** We can modify claw-freeness to claw-freeness of keys over the union \( \Xi^e \cup \Xi^d \). This would immediately imply xkcd, as \( Det \) can always return “No”. But in this case the reach of our feasibility result below will not extend to the KDM setting, since under KDM a single key is used in encryption and decryption and hence they always collide.

We will now state and prove our first feasibility result, which identifies a set of sufficient conditions on the pair \((\Xi^e, \Xi^d)\) so that the ideal cipher is KCA secure with respect to it.

**Theorem 1** (KCA security of the ideal cipher). Fix a key length \( k \) and block length \( n \) and let \( \Xi^e \) and \( \Xi^d \) be two sets of CDFs with signature \( \xi : \{0, 1\}^k \to \{0, 1\}^k \times \{0, 1\}^n \). Then for any \((\Xi^e, \Xi^d)\)-KCCCA adversary \( A \) against the ideal cipher making at most \( q \) direct ideal cipher queries, \( q_e \) encryption and \( q_d \) decryption queries, and for any detection algorithm \( Det \), there are (multi-shot) adversaries \( B_1, \ldots, B_5 \) such that

\[
\text{Adv}_{\text{Block}(k, n), \Xi^e, \Xi^d}(\Xi^e, \Xi^d, A) \leq \text{Adv}^{\text{m-kup}}_{\Xi^e}(B_1) + \text{Adv}^{\text{m-kup}}_{\Xi^d}(B_2) + \text{Adv}^{\text{m-cf}}_{\Xi^e}(B_3) + \text{Adv}^{\text{m-cf}}_{\Xi^d}(B_4)
\]

\[
+ \text{Adv}^{\text{m-xkcd}}_{\Xi^e, \Xi^d, \text{Det}}(B_5) + \left(q_{e} + q_{d} + q_{e}^2 + q_{d}^2 + q_{e}q_{d}\right)/2^n.
\]

Algorithm \( B_1 \) outputs two lists of sizes at most \( \ell_1 \leq q_e \) and \( \ell_2 \leq q_d \); \( B_2 \) outputs lists of sizes at most \( \ell_1 \leq q_e \) and \( \ell_2 \leq q_d \); \( B_3 \)'s lists are of sizes at most \( \ell_1 \leq q_e \); \( B_4 \)'s lists are of sizes at most \( \ell_1 \leq q_e \) and \( \ell_2 \leq q_d \).

**Proof.** The idea of the proof is that we will gradually modify KC-CCA of Fig. 4 so that KCEN and KCDEC oracles run with two independent and forgetful random oracles, denoted by \( \mathbb{S}^+ \) and \( \mathbb{S}^- \) respectively, and further the validity checks are performed by the detection algorithm \( Det \). These modifications ensure that we arrive at a game which does not depend on the explicit key-correlated keys, messages, or ciphertexts.

Pictorially, in this intermediate game with forgetful oracles the adversary can make queries that correspond to 1 (corresponding to \( iE \)), 2 (corresponding to \( iD \)), 3 and 4:

\[
\begin{array}{c}
1 \quad \text{\( iE \)} \\
2 \quad \text{\( iD \)} \\
\text{\( \mathbb{S}^+ \)} \\
\text{\( \mathbb{S}^- \)} \\
3 \quad \text{\( \mathbb{S}^- \)} \\
4 \quad \text{\( \mathbb{S}^- \)}
\end{array}
\]

We now describe the transitional steps; the details can be found in Figs. 5 and 6.

**Game\textsubscript{0}**: This is the KC-CCA game with respect to \( b = 1 \) and where the four oracles lazily sample the ideal cipher. The same ideal cipher is used in \( iE \) and \( iD \) as those used in KCEN and KCDEC oracles. Consistency is ensured via shared lists \( T^+, T^- \), \( \text{Im}^+ \), and \( \text{Im}^- \) used in lazy sampling.

**Game\textsubscript{1}**: This game differs from Game\textsubscript{0} in that different lists are used in \( iE \) and \( iD \) and in KCEN and KCDEC. The game still ensures consistency across the lists and hence there are no functional changes. This game also sets a flag \( \text{Bad}_{\text{kup}} \) preparing us to decouple the two lists. More precisely, whenever a call to \( iE \) or \( iD \) needs to use the lists of KCEN and KCDEC, or vice versa, \( \text{Bad}_{\text{kup}} \) is set. This game also sets a bad flag \( \text{Bad}_{\text{adi}} \) whenever it samples a ciphertext (resp. plaintext) that was sampled before and hence appears on \( \text{Im}^+_1 \) or \( \text{Im}^+_2 \) (resp. \( \text{Im}^-_1 \) or \( \text{Im}^-_2 \)).
Game2 : This game omits the code after $\text{Bad}_{\text{skdec}}$ or $\text{Bad}_{\text{col2}}$ is set. Thus Game1 and Game2 are identical until $\text{Bad}_{\text{skup}}$ or $\text{Bad}_{\text{col1}}$ is set. We defer the analysis of this event to a later game. As a result the two ideal ciphers used in $iE$ and $iD$ and $\text{KCEnc}$ and $\text{KCDec}$ are decoupled.

Game3 : This game checks for repeat queries to the ideal cipher used within the $\text{KCEnc}$ and $\text{KCDec}$ oracles. (Note that the queries to these oracles are assumed, without loss of generality, to be distinct.) The game sets a flag $\text{Bad}_i$ when this takes place. This game also sets a bad flag $\text{Bad}_{\text{col2}}$ whenever it samples a ciphertext (resp. plaintext) that was sampled before in a previous $\text{KCEnc}$ or $\text{KCDec}$ query (resp., also in a previous $\text{KCEnc}$ and $\text{KCDec}$ query). By dropping the code after $\text{Bad}_i$ and $\text{Bad}_{\text{col2}}$, we transition to a game where the ideal cipher within $\text{KCEnc}$ and $\text{KCDec}$ act as forgetful random oracles.

Game4 : This game uses the detection algorithm $\text{Det}$ to check whether or not queries to $\text{KCEnc}$ or those to $\text{KCDec}$ are valid. More explicitly:

1. For each decryption query $\xi^d$, oracle $\text{KCDec}$ runs $\text{Det}(\xi^c, C, \xi^d, 0^n, 1)$ for all previous queries $\xi^e$ to $\text{KCEnc}$, which were answered with $C$. (Intuitively, $\text{Det}$ checks whether $\xi^d(K) = (\xi^e|_1(K), C)$; the input $0^n$ is arbitrary.)

2. Analogously, for each query $\xi^e$, oracle $\text{KCEnc}$ runs $\text{Det}(\xi^e, 0^n, \xi^d, M, 2)$ for all previous queries $\xi^d$ to $\text{KCDec}$ which resulted in output $M$. If $\text{Det}$ (which intuitively checks if $\xi^e(K) = (\xi^d|_1(K), M)$) returns “Yes” for any of these, $\text{KCEnc}$ returns $\perp$, else it answers randomly.

Note that this game introduces no functional changes since it still computes the validity of a query using the lists. However it sets a bad flag $\text{Bad}_{\text{skdec}}$ when the validity check determined by $\text{Det}$ is different from the (true) value of validity determined using the lists $\text{ML}$ and $\text{CL}$. This then prepares us to drop the explicit check using the lists and only use $\text{Det}$ in the next game.

Game5 : This game drops the code after $\text{Bad}_{\text{skdec}}$, and so is identical to Game4 unless $\text{Bad}_{\text{skdec}}$ has been set.

Note that in Game5 the $iE$ and $iD$ oracles implement an ideal cipher whereas the $\text{KCEnc}$ and $\text{KCDec}$ oracles implement two forgetful random oracles, except that sometimes they might return $\perp$ if the $\text{Det}$ says that a query is invalid. To signify these, in what follows we refer to $\text{KCEnc}$ by $\mathcal{S}$ and $\text{KCDec}$ by $\mathcal{S}^-$. We also use $iE^+$ for $iE$ and $iE^-$ for $iD$.

Note also that the above games are all identical unless one of the bad flags is set. Thus we analyze the probability of setting any of the bad flags in the final game Game6, where the behavior of the oracles are independent of the secret key $K^*$.

We start by analyzing the probability of setting $\text{Bad}_{\text{skup}}$ and $\text{Bad}_{\text{col1}}$. Flag $\text{Bad}_{\text{skup}}$ can be set under $iE$ if a query $(K, M)$ to $iE$ appears on $T^+_2$. Since only $\text{KCEnc}$ and $\text{KCDec}$ write to $T^+_2$, this would be the case if (a) $(K, M) = \xi^e(K^*)$ for some query $\xi^e$ to $\text{KCEnc}$ or (b) $(K, M) = (\xi^d|_1(K^*), M')$ for $M'$ a randomly chosen output of $\text{KCDec}$.

In case (a) we can build an adversary that wins the multi-shot unpredictability of $\Xi^e$ by simulating the oracles as in Game5 (without the need for $K^*$, as all dependences on $K^*$ are eliminated by Game5) and outputing two lists $L_1$ and $L_2$ consisting of all tuples $(K, M)$ that were submitted to $iE$ and $\xi^e$ that were submitted to $\text{KCEnc}$. These lists are of sizes $q$ and $q_e$ respectively.

In case (b), however, we build an adversary that wins the multi-shot key unpredictability of $\Xi^d$ (note the exponent $d$). The four oracles are simulated as in Game5 (once again without the need for $K^*$), but now the adversary outputs two lists $L_1$ and $L_2$ consisting
Game\textsuperscript{0}.\textsuperscript{2}, A:\n\begin{align*}
b & \leftarrow 1; K^* \leftarrow \{0, 1\}^k \\
(iE, iD) & \leftarrow \text{Block}(K, n) \\
b' & \leftarrow A(iE, iD); KCenc; KCdec \\
\text{return} \ (b' = 1) \\
\end{align*}

Proc. \text{ie}(K, M):
\begin{align*}
\text{if } T^+[K, M] & : \text{return } T^+[K, M] \\
\text{else } C & \leftarrow \{0, 1\}^n \setminus \text{Im}^+[K] \\
T^+[K, M] & := C \\
\text{Im}^+[K] & \leftarrow \text{Im}^+[K] : C \\
\text{Im}^-[K] & \leftarrow \text{Im}^-[K] : M \\
\text{return } C
\end{align*}

Proc. \text{id}(K, C):
\begin{align*}
\text{if } T^-[K, C] & : \text{return } T^-[K, C] \\
\text{else } M & \leftarrow \{0, 1\}^n \setminus \text{Im}^-[K] \\
T^-[K, C] & := M \\
T^+[K, C] & := C \\
\text{Im}^-[K] & \leftarrow \text{Im}^-[K] : M \\
\text{Im}^+[K] & \leftarrow \text{Im}^+[K] : C \\
\text{return } M
\end{align*}

Proc. KCenc(ξ):
\begin{align*}
(K, M) & \leftarrow ξ^*(K^*) \\
\text{if } (K, M) & \in \text{ML} : \text{return } ⊥ \\
\text{if } T^+[K, M] & : \text{return } T^+[K, M] \\
\text{else } C & \leftarrow \{0, 1\}^n \setminus \text{Im}^+[K] \\
T^+[K, M] & := C \\
T^-[K, C] & := M \\
\text{Im}^+[K] & \leftarrow \text{Im}^+[K] : C \\
\text{Im}^-[K] & \leftarrow \text{Im}^-[K] : M \\
\text{CL} & \leftarrow \text{CL} : (K, C) \\
\text{return } C
\end{align*}

Proc. KCdec(ξ):
\begin{align*}
(K, C) & \leftarrow ξ^*(K^*) \\
\text{if } (K, C) & \in \text{CL} : \text{return } ⊥ \\
\text{if } T^-[K, C] & : \text{return } T^-[K, C] \\
\text{else } M & \leftarrow \{0, 1\}^n \setminus \text{Im}^-[K] \\
T^-[K, C] & := M \\
T^+[K, M] & := C \\
\text{Im}^+[K] & \leftarrow \text{Im}^+[K] : M \\
\text{Im}^-[K] & \leftarrow \text{Im}^-[K] : C \\
\text{ML} & \leftarrow \text{ML} : (K, M) \\
\text{return } M
\end{align*}

Figure 5: Game\textsuperscript{0}, Game\textsubscript{1}, and Game\textsubscript{2}.  

Aisling Connolly, Pooya Farshim, and Georg Fuchsbauer 13
Figure 6: Game\textsubscript{3}, Game\textsubscript{4}, and Game\textsubscript{5}.
of all keys $K$ for tuples $(K, M)$ submitted to $\text{iE}$ and $\xi^e$ that were submitted to $\text{KCDec}$. These lists are of sizes $q_e$ and $q_d$ respectively.

We now look at the probability of setting $\text{Bad}_{\text{col}_1}$. A straightforward analysis shows that this probability is upper bounded by $q(q_e + q_d)/2^n$.

The probability of setting $\text{Bad}_{\text{up}}$ under $\text{iD}$ is analyzed similarly to cases (a) and (b) above via reductions to the multi-shot unpredictability of $\Xi^d$ and the multi-shot key unpredictability of $\Xi^e$. The probability of setting $\text{Bad}_{\text{col}_1}$ under $\text{iD}$ is at most $q(q_e + q_d)/2^n$.

Next we consider the probability of setting $\text{Bad}_{\text{cf}}$ or $\text{Bad}_{\text{col}_1}$. Flag $\text{Bad}_{\text{cf}}$ can be set under $\text{KCEnc}$ as a result of a query $\xi_0^c$ if either (a) a previous distinct query $\xi_1^c$ to $\text{KCEnc}$ resulted in $\xi_0^c(K^*) = \xi_1^c(K^*)$, or (b), a previous query $\xi_1^c$ to $\text{KCDec}$ resulted in $\xi_0^c(K^*) = (\xi_1^c|_{\text{det}}(K^*), M)$, where $M$ is a random value chosen by $\text{KCDec}$ in a previous query.

In case (a) by collecting the list of $\xi^c$ queried to $\text{KCEnc}$ we can break the multi-shot claw-freeness of $\Xi^e$ (simulating the oracles as in $\text{Game}_{\text{ci}}$). For case (b) to take place it must be that $\text{Det}$ classified the query as valid. However, this leads to setting $\text{Bad}_{\text{sked}}$ since $M$ was returned from the $\text{KCDec}$ oracle and hence $(\xi_1^c|_{\text{det}}(K^*), M)$ appears on ML but $\text{Det}$ did not detect this. We analyze the probability of setting $\text{Bad}_{\text{sked}}$ below. The probability of setting $\text{Bad}_{\text{col}_1}$ in $\text{KCDec}$ can be upper bounded by $q_e(q_e + q_d)/2^n$.

The probability of setting $\text{Bad}_{\text{cf}}$ under $\text{KCDec}$ is analyzed as above by a reduction to the multi-shot claw-freeness of $\Xi^d$ (or setting $\text{Bad}_{\text{sked}}$). The probability of setting $\text{Bad}_{\text{col}_2}$ in $\text{KCDec}$ is upper bounded by $q_d(q_e + q_d)/2^n$.

Finally we consider the probability of setting $\text{Bad}_{\text{sked}}$. Flag $\text{Bad}_{\text{sked}}$ is set exactly when $\xi^c(K^*)$ appears on ML as a result of a query $\xi^d$ to $\text{KCDec}$ with response $M$ and $\text{Det}$ says it does not, or that it does not and $\text{Det}$ says it does. We can bound the probability of setting $\text{Bad}_{\text{sked}}$ under $\text{KCDec}$ by a reduction to the multi-shot xkd property by collecting all queries $\xi^c$ made to $\text{KCEnc}$ and all query-answer pairs $(\xi^d, M)$ made to and received form $\text{KCDec}$ in two lists and outputting them lists in the multi-shot xkd game. The probability of setting of $\text{Bad}_{\text{sked}}$ under $\text{KCDec}$ is analyzed analogously.

The theorem now follows by putting the bounds established above together.

The above theorem shows that key-unpredictability, claw-freeness, and xkd are sufficient for achieving KCA security in the ideal-cipher model. This raises the question whether or not they are also necessary. Key-unpredictability is. Any adversary that can predict the key output of $\xi^c$ can be used to win the KC game as follows: (1) ask for an encryption under $\xi^c(K^*)$ to get a ciphertext $C$; (2) predict $\xi^c|_{\text{det}}(K^*)$ as $K$; (3) compute $\text{iD}(K, C)$ to get $M$; (4) compute $\text{iE}(K, M)$ to get $C'$; (5) return $b' := 1$ if $C = C'$ and $b' := 0$ otherwise. It is easily seen that this attack returns $b' = 1$ with the same probability of predicting keys when $b = 1$. On the other hand, it only succeeds with negligible probability when $b = 0$ as the encryption oracle and the challenge oracle use independent ideal ciphers. We deal with such attacks in case (b) of ($\xi^c$, $\text{iE}^c$) in the proof.

Claw-freeness, on the other hand, is not necessary. Consider a CDF set consisting of exactly two functions that collide with probability 1/2, e.g., $K \mapsto K$ and $K \mapsto \min(K, \overline{K})$. Then a security proof for the ideal cipher can be easily given: guess with probability 1/2 if $\min(K, \overline{K}) = K$. Next depending on the guess simulate the key-correlated oracles either using the same set of ideal cipher oracles or independent ones. However, one can also exhibit CDF sets that are not claw-free and do lead to a KCA on the ideal cipher. Indeed, any set of $k + 1$ functions that has claws with the identity function depending on whether or not $K_i$, the $i$-th bit of the key, is 1 can be used to recover the key one bit at a time. In practice this translates to recovering the key by observing the repetition patterns in the outputs.

\footnote{The above cases take care of collisions between wires (1/2, 3), (1/2, 4) in the picture above.}
Finally, there is also a pair of CDF sets which are not xkcd and can be used to attack the ideal cipher. Let $\xi^d(K) := (K, 0^n)$ and $\xi^d[i, C](K) := (K \oplus (K_i)^n =: K', C)$, that is, $K' = \overline{K}$ if $K_i = 1$ and $K' = K$ otherwise. Let $C$ be the answer to $\text{KCDec}(\xi^d)$. Now, for $i = 1, \ldots, k$, by checking whether $\text{KCDec}(\xi^d[i, C])$ returns $\perp$ or not, one can again recover the key one bit at a time. In practice this translates to observing the equality pattern of the decrypted ciphertexts with $0^n$.

Theorem 1 and Proposition 1 can be used to recover known feasibility results for RKA and KDM security of the ideal cipher as follows.

**RKA security.** An RKA-only CDF set $\Xi$ associated to a set $\Phi$ is key-unpredictable iff $\Phi$ is unpredictable in the sense of [BK03]. Indeed, a predictor for a related-key derivation function $\phi$ can be converted to a key-predictor for $\xi: K \mapsto (\phi(K), M)$ for any fixed $M$.

Conversely, a key-predictor for $(\phi(K), M)$ in particular predicts $\phi(K)$. Similarly, $\Xi$ is claw-free iff $\Phi$ is claw-free in the sense of [BK03]. A claw in $\Phi$ can be immediately converted to a claw in $\Xi$ (by adding any message $M$). Conversely, a claw $((\phi_1(K), M_1), (\phi_2(K), M_2))$ is necessarily also a claw between $\phi_1$ and $\phi_2$. For the (funny) xkcd property, $\text{Det}$ given $((\phi_1, M_1), C, (\phi_2, C_2), M, i)$ needs to check if (1) $\phi_1(K) = \phi_2(K)$ and (2) if $C = C_2$ when $i = 1$ or if $M = M_1$ when $i = 2$. By claw-freeness, (1) would only occur if $\phi_1 = \phi_2$; the second condition on the other hand is trivial to check. Thus, claw-freeness implies xkcd.

**KDM security.** A KDM-only $\Xi$ associated to a KDM set $\Psi$ is always key-unpredictable with single-shot advantage $1/2^n$. This is because KDM-only functions leave the random key unmodified, which remains information-theoretically hidden from the adversary in the key-unpredictability game. Moreover, and similarly to the RKA case, the set $\Xi$ is claw-free iff $\Psi$ is. Once again, this is because KDM-only CDFs do not modify the key component and hence any claw must be on the message part. For the xkcd property, given $((id, \psi_1), C, (id, C_2), M, i)$, algorithm $\text{Det}$ needs to check (1) if $id(K) = id(K)$, which always holds; and (2) if $C_2 = C$ when $i = 1$ or if $\psi_1(K) = M$ when $i = 2$. The first of these is trivially checkable. For the second, two cases could occur: either $\psi_1(K)$ is a constant function mapping to $M_1$, in which case $\text{Det}$ can easily check if $M_1 = M$; or $\psi_1$ is non-constant. However, since $\Psi$ contains all constant functions, $\psi_1$ and $M$ can be used to break claw-freeness. Hence in this case $\text{Det}$ rejects. So similarly to the RKA setting, claw-freeness implies xkcd.

**KCA security.** We now show that the reach of the above feasibility result for the ideal cipher extends beyond RKA and KDM security. Suppose $k = n$ and recall the set

$$\Xi^0 := \left\{ K \mapsto (K \oplus \Delta_1, \alpha \cdot K \oplus \Delta_2): \Delta_1, \Delta_2 \in M, \alpha \in \{0, 1\} \right\}. \tag{4}$$

This set captures related keys that are computed as offsets, key-independent messages when $\alpha = 0$, and key-dependent messages which are offsets of the key when $\alpha = 1$. We identify functions in this set with tuples $(\Delta_1, \Delta_2, \alpha)$.

This set is key-unpredictable with single-shot advantage at most $1/2^n$, since xor-ing with $\Delta_1$ acts as a permutation and $K$ remains information-theoretically hidden. It is also claw-free with single-shot advantage at most $1/2^n$: if the offsets $\Delta_1$ and $\Delta_1'$ for two functions $\xi, \xi'$ are different, there are no claws. If they are the same, and $\alpha = \alpha'$, then a claw also implies $\Delta_2 = \Delta_2'$, making the two functions identical. So $\alpha \neq \alpha'$ is necessary for claws, in which case the functions collide on a random key with probability $1/2^n$.

The set is also xkcd with single-shot advantage $2/2^n$ with respect to the following detector (recall that $\text{Det}(\xi^c, C, \xi^d, M, 1)$ should check whether $(\xi^c|_1(K), C) = \xi^d(K)$):

$$\text{Det}_\Xi((\Delta_1^d, \Delta_2^d, \alpha^d), C, (\Delta_1^d, \Delta_2^d, \alpha^d), M, i):$$

if $i = 1$: if $\Delta_1^d = \Delta_1^c \land \alpha^d = 0 \land \Delta_2^d = C$ then return “Yes”

if $i = 2$: if $\Delta_1^d = \Delta_1^c \land \alpha^d = 0 \land \Delta_2^d = M$ then return “Yes”

else return “No” \tag{5}
Consider case \( i = 1 \) when \( \text{Det_3} \) is trying to decide if \((K \oplus \Delta_1, C) = (K \oplus \Delta_1', \alpha_d, K \oplus \Delta_2')\). If \( \text{Det_3} \) returns “Yes”, then both the key components and the ciphertext components collide; the answer is thus correct. Suppose it returns “No”. If \( \Delta_1' \neq \Delta_1 ' \), the key components do not collide and the answer is correct. Suppose \( \Delta_1' = \Delta_2' \): if \( \alpha_d = 1 \), the probability that \( \alpha_d \cdot K \oplus \Delta_1 ' = C \) is \( 1/2^n \) since \( \oplus \) acts a permutation and \( K \) is random; if \( \alpha_d = 0 \), then \( \text{Det_3} \) checks if \( \Delta_2 ' = C \) and hence its answer is correct. Case \( i = 2 \) is dealt with similarly and we obtain overall single-shot advantage of at most \( 2/2^n \).

Theorem 1 thus implies that the ideal cipher is \((\Xi^\oplus, \Xi^\oplus)\)-KC-CCA secure.

6 KCA Security of 3-Round Even–Mansour

In this section we show that in contrast to the two-round case the three-round Even–Mansour cipher with reuse of keys and independent permutations \( \text{EM}^{p_1, p_2, p_3}[K, K, K, K] \), defined in Sec. 4.2, is \((\Xi^\oplus, \Xi^\oplus)\)-KC-CCA secure, with \( \Xi^\oplus \) as defined in (4). We note that, for \( \Phi^\oplus, \Psi^\oplus \) as in (2), this construction is known to be both \( \Phi^\oplus\)-RK-CCA and \( \Psi^\oplus\)-KDM-CCA secure [FP15, FKV17]. We build on these works to show KC-CCA security in the theorem.

**Theorem 2** (KCA security of 3-round EM). \( \text{EM}^{p_1, p_2, p_3}[K, K, K, K] \) is \((\Xi^\oplus, \Xi^\oplus)\)-KC-CCA secure in the random-permutation model for \( p_2 \). More precisely, for any adversary \( A \) against the \((\Xi^\oplus, \Xi^\oplus)\)-KC-CCA security of \( \text{EM}^{p_1, p_2, p_3}[K, K, K, K] \) we have

\[
\text{Adv}_{\text{EM}^{p_1, p_2, p_3}[K, K, K, K]}^{\text{kc-cca}}(\Xi^\oplus, \Xi^\oplus, A) \leq 2 \cdot (q_{\oplus} + q_{\ominus} + q_e^2 + q_d^2 + q_e q_d)/(2^n - q),
\]

where \( q \) is the maximum number of queries of \( A \) to \( p_2^\pm \) (over all \( i = 1, 2, 3 \)) and \( q_{\oplus} \) and \( q_{\ominus} \) are the maximum number of encryption and decryption queries of \( A \), respectively.

**Proof.** To analyze the KC-CCA security of Even–Mansour, our strategy will be similar to that in Theorem 1 and those in [FP15, FKV17]. As in these proofs, we replace the last-round permutation \( p_2^+ \) in KCENC queries with a forgetful random oracle and also replace the first-round permutation \( p_1^- \) in KCDEC with another forgetful random oracle. After these replacements the outputs of KCENC and KCDEC will be fully randomized and independent of the inputs; the game is thus independent of the challenge bit \( b \).

When simulating KCENC and KCDEC with forgetful oracles, we still need to ensure that illegal queries are answered correctly with \( \bot \). As in the proof of Theorem 1, we do this using a detector. In particular, we use \( \text{Det}_3 \) defined in (5), which on input \( ((\Delta_1, \Delta_2, \alpha), C, (\Delta'_1, \Delta'_2, \alpha'), M, 1) \) returns “Yes” if \( \Delta_1 = \Delta_1' \) and \( \alpha' = 0 \) and \( \Delta_2 ' = C \) (and “No” otherwise); and on input \( ((\Delta_1, \Delta_2, \alpha), C, (\Delta'_1, \Delta'_2, \alpha'), M, 2) \) returns “Yes” if \( \Delta_1 = \Delta_1' \) and \( \alpha = 0 \) and \( \Delta_2 = M \).

1. For each decryption query \( \xi^d \), oracle KCDEC runs \( \text{Det}_3(\xi^d, C, \xi^d, 0^n, 1) \) for all previous queries \( \xi^e \) to KCENC, which were answered with \( C \). If \( \text{Det} \) says “Yes” for any of these, KCDEC returns \( \bot \), else it outputs a random value from the domain.

2. For each encryption query \( \xi^e \), oracle KCENC runs \( \text{Det}_3(\xi^e, 0^n, \xi^d, M, 2) \) for all previous queries \( \xi^d \) to KCDEC, which were answered with \( M \). If \( \text{Det} \) says “Yes” for any of these, KCENC returns \( \bot \), else it answers with a random value.

These replacements, however, raise the question how \( p_2^+ \) in KCENC and \( p_3^- \) in KCDEC are treated (since their respective counterparts in KCDEC and KCENC were replaced). Previous works simply leave these oracles unchanged. Our proof diverges in this aspect, because the key-dependency of both keys and messages gives rise to queries that cannot be guaranteed to be disjoint from the forgetful oracle, or discarded as trivial/illegal as in the
We now bound the probabilities of these events.

For the KCA setting we will replace $P_3^-$ in KCDec (and analogously for $P_1^+$ in KCEnc) as follows. When Det$_\xi$ says “No” on a query $\xi^d = (\Delta_1^d, \Delta_2^d, \alpha^d)$, the game checks if each previous query $\xi = (\Delta_1, \Delta_2, \alpha)$ to KCEnc with output value $C$ satisfies
\[ C \oplus \Delta_1^d = \Delta_2^d \oplus \Delta_1^d. \]

If so, oracle $\bar{P}_3^-$ outputs the input $U$ to $\$^+$ that was used to compute $C$. If multiple values are found, $\bar{P}_3^-$ returns $\bot$. If the latter is not the case, $\bar{P}_3^-$ is sampled as an independent permutation and consistent with $P_3$. Permutation $P_3^\pm$, when directly queried, is also sampled consistently with $\bar{P}_3^-$ on all those entries that are not kept consistent with $\$^+$. More precisely, oracles $\$^+, P_3^\pm$ and $\bar{P}_3^-$ write to independent lists $L_1$, $L_2$ and $L_3$. Oracle $\$^+$ checks consistency with none of the lists, oracle $P_3^\pm$ checks consistency with $L_2$ and $L_3$ and oracle $\bar{P}_3^-$ checks consistency with $L_1$, $L_2$ and $L_3$.

Oracle $P_1^+$ in KCEnc is replaced with $P_1^+$ similarly by checking if $M \oplus \Delta_1^d = \Delta_2^d \oplus \Delta_1^d$ and outputting the input $U$ to $\$^-$ that was used to compute $M$.

As a result of the above replacements, the following inconsistencies could arise:

(a) Identical queries to a permutation oracle are answered differently due to a replacement.

(b) Different queries to a permutation oracle and its replacement are answered identically (and hence do not respect permutativity).

(c) Multiple candidates for $U$ are found when sampling $\bar{P}_3^-$ or $P_3^+$.

We now bound the probabilities of these events.

**Multiple candidates** This event will happen due to: (1) a collision in the outputs of $\$^+$, which happens with probability $1/2^n$ for each pair of queries to KCEnc; or (2) a collision in the output of $\$^+$ and an output entry in $P_3^\pm$. This event is equivalent to a collision in the outputs of $\$^+$ and $P_3^\pm$, which pertains to case (b) and is analyzed next.

($\$^+, P_3^\pm$) A direct $P_3^\pm$ call in either direction collides with a point queried to $P_3^+$ due to a query $\xi^d = (\Delta_1^d, \Delta_2^d, \alpha^d)$ to KCEnc. Suppose the direct query results in $V = P_3^\pm(U)$ or $U = P_3^\pm(V)$. There are now two cases. (a) The inputs collide, i.e.,
\[ P_2(P_1(\Delta_1^d \oplus \Delta_2^d \oplus (1-\alpha) \cdot K^*) \oplus \Delta_1^d \oplus K^*) \oplus \Delta_1^d \oplus K^* = U. \]

If $Y := P_1(\Delta_1^d \oplus \Delta_2^d \oplus (1-\alpha) \cdot K^*) \oplus \Delta_1^d \oplus K^*$ was queried to $P_2$ then the adversary can compute $K^* = P_2(Y) \oplus \Delta_1^d \oplus U$, which is only possible with probability $1/2^n$, since the key remains information-theoretically hidden. If it was not queried, the probability that (6) holds is also $1/(2^n - q)$, since $P_2$ is a random permutation, where outputs are chosen in a set of size at least $2^n - q$.

The other case is that (b) the outputs collide. If the $\$^+$ query comes after the direct query, the probability of this event is $1/2^n$. Now suppose the $\$^+$ query comes first and outputs $C$; thus due to the collision we have that $C = V \oplus K^* \oplus \Delta_1$. But in this case the adversary can compute $K^* = V \oplus \Delta_1$, which happens with probability at most $1/2^n$ (as the key remains information theoretically hidden).

($\$^-, P_3^\pm$) A direct $P_3^\pm$ call in either direction collides with a point queried to $\$^-$ due to a query $\xi^d$ to KCDec. This event is analyzed analogously to case ($\$^+, P_3^\pm$) above.

($\$^+, \$^+$) The are two cases: (a) two inputs collide or (b) two outputs collide. In case (a) two distinct calls $\xi^d = (\Delta_1^d, \Delta_2^d, \alpha^d)$ and $(\xi^d)' = (\Delta_1', \Delta_2', \alpha')$ to the KCEnc oracle...
result in querying $\$^+$ on the same input. For this event to occur we must have that

$$P_2 \left( \frac{\partial_1 \oplus \partial_2 \oplus (1 - \alpha) \cdot K^* \oplus \partial_1 \oplus K^*}{\partial_1 \oplus \partial_1} \right) = Y \quad (6)$$

$$= X \quad (7)$$

If $Y = Y'$ then $P_1(X) \oplus \partial_1 = P_1(X') \oplus \partial_1'$ and from (7) we get $\partial_1 = \partial_1'$. Since $P_1$ is a permutation, this implies that $X = X'$. If further $\alpha = \alpha'$ then we would also have $\partial_2 = \partial_2'$, which means $\xi^c = (\xi^c)'$, contradicting distinctness of queries. Thus $\alpha \neq \alpha'$ and hence $\partial_2 \oplus \partial_2' = K^*$ (since $X = X'$), which only happens with probability $1/2^n$ as it leads to guessing the key.

Assume now that $Y \neq Y'$.

- If the adversary does not query $P_2$ on either $Y$ or $Y'$ then $P_2(Y)$ or $P_2(Y')$ would be distributed randomly and independently of the adversary’s view. Hence the probability that the above equality holds would be at most $1/(2^n - q)$ for each pair of queries.

- Suppose both $Y$ and $Y'$ are queried to $P_2$. Then $P_1$ must have been queried on both $X$ or $X'$, since otherwise the probability of computing $Y$ or $Y'$ would be at most $1/(2^n - q)$. If both $X$ and $X'$ are queried to $P_1$ and $\alpha = 0$ or $\alpha' = 0$, then for each query the probability of querying $X$ (or $X'$) to $P_1$ is $1/2^n$, because $K^*$ is information-theoretically hidden. The remaining case is $\alpha = \alpha' = 1$. Since the adversary knows $Y$ (it was queried to $P_2$), it can compute $K^* = Y \oplus P_1(\partial_1 \oplus \partial_2) \oplus \partial_1$. Again, since $K^*$ is information-theoretically hidden, this can only happen with probability $1/2^n$.

In case (b) outputs of $\$^+$ collide with probability $1/2^n$ for each pair of queries.

$(\$^-, \$^-)$ Two distinct $(\xi^d, (\xi^d)') \in \Xi^d$ to KCDec result in querying $\$^-$ on the same point. This event is analyzed analogously to case $(\$^+, \$^+)$ above.

$(\$^+, \tilde{P}_\$^+)$ Call $(\xi^c, (\xi^c)') \in \Xi^c$ to KCEnc and $(\xi^a, (\xi^a)') \in \Xi^a$ to KCDec result in (a) querying $\tilde{P}_\$^+ \oplus \partial_1 \oplus K^*$ on an input matching an output of $\$^+$; or in (b) querying $\tilde{P}_\$^- \oplus \partial_1 \oplus K^*$ on an input whose output matches an input of $\$^+$. Let $C$ be the reply to query $\xi^c$. Then the value chosen by $\$^+$ is $C \oplus K^* \oplus \partial_1$. Thus case (a) would happen if

$$C \oplus K^* \oplus \partial_1 = \alpha \cdot K^* \oplus \Delta_1 \oplus \Delta_1.$$

If $\alpha^d = 1$, then

$$C \oplus K^* \oplus \partial_1 = \Delta_1 \oplus \Delta_1,$$

which leads to guessing $K^*$ and happens with probability at most $1/2^n$. On the other hand, if $\alpha^d = 0$, we would have that

$$C \oplus \partial_1 = \Delta_1 \oplus \Delta_1.$$

But for such queries either $\tilde{P}_\$^- is sampled to be consistent with $\$^+$ (or it is not called at all when Det returns “Yes”).

Let us now look at case (b), which would happen if

$$= Y$$

$$\quad = X$$

$$\quad = \tilde{P}_\$^- \left( \Delta_1 \oplus \Delta_2 \oplus (1 - \alpha) \cdot K^* \right) \oplus \Delta_1 \oplus K^* \quad (8)$$
If $\Delta^+ \oplus \Delta^i \oplus (1 - \alpha^d) \cdot K^*$ was directly queried to $P^-_3$, an inconsistency of type $(\xi^+; P^+_3)$ arises, which we dealt with above. On the other hand, it could be that this value was never queried to $P^-_3$.

If the KCENCC query comes before KCDECC and the inputs are distinct—colliding inputs was case (a) above—$\tilde{P}^-_3$ chooses a random value, which collides with the input to $\xi^+$ with probability at most $1/(2^n - q)$.

Suppose now that the KCENCC query comes after KCDECC. Suppose the input to KCENCC is $\xi^+ = (\Delta^+_1, \Delta^+_2, \alpha^e)$. The value computed as input to $\xi^+$ is independent of the output of $\tilde{P}^-_3$ unless $P^+_1$ is kept consistent with $\xi^+$ (since otherwise $P_1$ and $P_2$ are independent of $P^-_3$). If independent, collisions happen with probability at most $1/(2^n - q)$. Otherwise, dependency implies that $M \oplus \Delta^+_i = \Delta^1_i \oplus \Delta^2_i$ where $M$ is the output of KCDECC. This query will be marked as illegal unless $\Delta^+_i \neq \Delta^1_i$. In this case the adversary has found a collision of the form

$$P_2(Y' \oplus K^* \oplus \Delta^+_1) \oplus K^* \oplus \Delta^1_i = U,$$

where $Y' = P^+_2(U \oplus K^* \oplus \Delta^+_1) \oplus K^* \oplus \Delta^1_i$ is the value input to $\xi^+$ and output by $P^+_1$ due to the consistency check in KCENCC. Rearranging and applying $P^-_2$ to the above equality yields $Y' \oplus K^* \oplus \Delta^+_2 = P^-_2(U \oplus K^* \oplus \Delta^+_1)$. Substituting the expression for $Y'$ we get that $P^+_2(U \oplus K^* \oplus \Delta^+_1) \oplus \Delta^1_i \oplus \Delta^+_i = P^-_2(U \oplus K^* \oplus \Delta^+_1)$. Since $P^-_2$ is a permutation, this would happen iff $\Delta^+_i = \Delta^1_i$, which is an illegal query.

$(\xi^-, P^+_1)$ A call $\xi^d$ to KCDECC and a call $\xi^e$ to KCENCC result in querying $\xi^-$ and $P^+_1$ on the same point. This event is analyzed analogously to case $(\xi^+, P^-_3)$ above.

$(\tilde{P}^-_3, P^+_3)$ A call $\xi^d$ to KCDECC and a direct query to $P^+_3$ are such that either the inputs or the outputs collide. If the KCDECC query comes after the direct query, since oracle $\tilde{P}^-_3$ is sampled to be consistent with $P_3$, no inconsistency arises. If $P_3$ comes after KCDECC then an inconsistency arises only with an entry on $P^-_3$ that is also on $\xi^+$. But in this case an inconsistency of the form $(\xi^+, P^+_3)$ has been found.

$(\tilde{P}^-_1, P^+_1)$ A call $\xi^e$ to KCENCC and a direct query to $P^+_1$ collide. This case is treated analogously to $(\tilde{P}^-_3, P^+_3)$ above.

KCENCC inconsistently rejects If $\text{Det}_\xi$ does not output an incorrect answer, $\perp$ is the correct answer. On the other hand, as shown at the end of previous section, $\text{Det}_\xi$ with $i = 1$ outputs an incorrect answer with probability at most $1/2^n$ per inputs checked.

KCDECC inconsistently rejects Similarly to the above, this event occurs with probability at most $1/2^n$ per inputs checked.

The theorem follows by putting the above bounds together.

Remark. A potential strengthening of the above theorem would be to consider a wider class of KCA attacks for which the ideal cipher is known to be secure. One possibility would be the class of key-unpredictable, claw-free, and xkcd-secure CDF sets as shown in Theorem 1. Although feasibility of this level of security claim for iterated Even–Mansour ciphers would follow from known indifferentiability [LS13, DSST17], this would require a larger number of rounds and also comes at the cost of lower levels of security. It remains an interesting open question to find the minimal number of rounds needed in the Even–Mansour ciphers that yields $(\Xi^*, \Xi^d)$-KC-CCA security with respect to any pair $(\Xi^*, \Xi^d)$ for which the ideal cipher is also $(\Xi^*, \Xi^d)$-KC-CCA secure.

Remark. We note that our model does not consider the possibility of preprocessing attacks, and establishing security bounds based on the size of non-uniform advice and the number of on-line queries remains an interesting line of research for future work.
7 Authenticated Encryption

We have seen that it is possible to achieve KCA-secure blockciphers. Given this, we now turn our attention to higher-level primitives and investigate the possibility of constructing KCA-secure primitives which rely on blockciphers. A natural first step is to explore symmetric encryption. Specifically, we consider the notion of authenticated encryption with associated data (AEAD).

AE SYNTAX. An AEAD scheme is a 3-tuple of algorithms $SE = (Gen, Enc, Dec)$, where (1) $Gen$ is the randomized key generation algorithm that returns a bit string $K$ from a finite key-space $K$ (which is typically $\{0, 1\}^k$ for some $k \in \mathbb{N}$). (2) $Enc : K \times M \times N \times H \rightarrow C$ is the deterministic encryption algorithm which takes as input a key $K$, a message $M$ from message space $M$, a nonce $N$ from nonce space $N$, and possibly some associated header data $H$ from header-space $H$ and returns a ciphertext $C$ from some ciphertext-space $C$. We write this as $C \leftarrow Enc(K, M, N, H)$. (3) The deterministic decryption algorithm $Dec : K \times C \times N \times H \rightarrow M \cup \{\bot\}$ takes a key $K$, a ciphertext $C$, a nonce $N$, and possibly some associated header data $H$ and returns either a plaintext $M$ or the failure symbol $\bot \in M$. Correctness requires that $Dec(K, Enc(K, M, N, H), N, H) = M$ for all values of the inputs above.

AE SECURITY. We define the AE-security of $SE = (Gen, Enc, Dec)$ by considering the game described on the left in Fig. 7. The AE advantage of an adversary $A$ against an AEAD scheme is defined by

$$Adv_{AE}^SE(A) := 2 \cdot Pr[AE_{SE}^A] - 1.$$

We require that $A$ is nonce-respecting in that it does not repeat nonces in its encryption queries (but it may repeat them in decryption or across encryption and decryption queries).

KC-AE SECURITY. Let $SE$ be an AEAD scheme as defined above. Let $\Xi^e$ and $\Xi^d$ be CDF sets with signatures $\xi^e : K \rightarrow K \times M \times N \times H$ and $\xi^d : K \rightarrow K \times N \times H$ respectively. As is in the blockcipher case, $\Xi^e$ is used to generate correlated keys and key-correlated messages. In addition to this, for authenticated encryption, we allow all inputs to the encryption algorithm to be key-correlated. Unlike the blockcipher case however, we restrict the use of $\Xi^d$ to generate correlated keys, nonces, and headers as it seems unnatural to derive a key-correlated ciphertext for decryption. If $\xi^e \in \Xi^e$ and $\xi^d \in \Xi^d$ we note that the syntactical difference to $SE$ when considering KC-AE security is that the encryption oracle in the scheme only takes $\xi^e$ as input, while the decryption oracle takes $(\xi^d, C)$ as described in the KC-AE$^{\Xi^e, \Xi^d, A}$ game on the right in Fig. 7. The KC-AE advantage of an adversary $A$ against AEAD is defined by

$$Adv_{KC-AE}^{SE}(\xi^e, \xi^d, A) := 2 \cdot Pr[KC-AP_{SE}^{\Xi^e, \Xi^d, A}] - 1.$$

We require that nonces are not repeated in KC$\text{Enc}$ queries.

Security against key-correlated attacks is a strengthening of the standard notion of AE-security for a symmetric encryption scheme. As the syntax shows, the input to SE consists of $K, M, N,$ and $H$. A natural question may arise as to which inputs we allow to be key-correlated. Although we would like to provide guarantees against any correlated inputs, there are two immediate issues with header security. It can (sometimes) be the case that headers are not private, and if they are heavily key-correlated, an adversary could trivially retrieve the key. For this reason, we require that header data is private. But a more subtle point, as shown in [BK11], is that even with private headers, it is possible for an adversary to perform a full key-recovery attack by taking advantage of the pattern of errors returned while list-checking during the decryption process which may be key-dependent. This attack does not apply in our setting, as we return only one
The Appendix B) that any variant of the 22 Security of Symmetric Primitives against Key-Correlated Attacks be still able to obtain header security, but possibly requiring the key-claw-freeness of correlation-derivation messages of the same length. Decryption always returns the error symbol, except in the case the $i$ (indicating user number), and returns encryptions under a key to choose the number of keys to target, analogous to the number of users in the scheme. 

When headers or nonces are public—formally this means revealing the header $H$ or $N$ under the KCENC and KCDEC oracles—we can achieve security by requiring that headers or nonces are key-uncorrelated in the sense that their values can be computed from the description of input CDFs without the need for $K^*$. Looking ahead, this requirement would allow a proof strategy similar to one below in the private setting to be also applied in the public setting, with the difference that the values of headers and nonces would now be computed from the descriptions of the input CDFs.

The HwN Transform. We introduce the Hash-with-Nonce (HwN) transform with the goal of achieving KCA-security for symmetric encryption. Working in the random-oracle model, HwN converts a conventional AE-secure AEAD scheme $SE$ to a new one, $SE = HwN[SE, Gen] = (Gen, Enc, Dec)$, which we will prove to be KC-AE secure. $Gen$ is its key-generation algorithm, its key length is $k$ and nonce length is $n$. Its encryption and decryption algorithms are defined as follows, where $H : K \times \mathcal{N} \rightarrow K$ is a random oracle.

$$
\begin{align*}
\text{Enc}(K, M, N, H) : & \quad C \leftarrow Enc(H(K|N), M, N, H) \\
& \text{return } C
\end{align*}
$$

$$
\begin{align*}
\text{Dec}(K, C, N, H) : & \quad M \leftarrow Dec(H(K|N), C, N, H) \\
& \text{return } M
\end{align*}
$$

To show that $SE$ is $(\Xi^e, \Xi^d)$-KC-AE secure it will be convenient to reduce to a multi-user variant of the AE game. Multi-user authenticated encryption (MUAE) allows an adversary to choose the number of keys to target, analogous to the number of users in the scheme. In addition to taking a message, nonce, and header, the encryption oracle takes an index $i$ (indicating user number), and returns encryptions under a key $K_i$, or a random string of the same length. Decryption always returns the error symbol, except in the case the real algorithms are being used where the message $M$ is returned by decrypting $C$ under $K_i$. The adversary cannot ask user $i$ to decrypt a ciphertext that it previously obtained from this user. In a scheme with $n$ users, it follows via a simple hybrid argument (see Appendix B) that any AE-secure AEAD scheme is also MUAE secure.

The convenience in using MUAE lies in the fact that we can leverage the programmability of random oracles to implicitly map the hash values $H(\xi^*_i|3(K^*))$ of the $i$-th query $\xi^*_i$ to KCENC, to a new randomly and independently chosen key in the MUAE game. This also decouples the key used in encryption (i.e., $H(\xi^*_i|3(K^*))$) from the key-correlated messages $\xi^*_i|3(K^*)$, key-correlated nonces $\xi^*_i|3(K^*)$, and key-correlated headers $\xi^*_i|4(K^*)$.

Even if we define a model with different symbols to denote different types of decryption failures, we may be still able to obtain header security, but possibly requiring the key-claw-freeness of correlation-derivation functions.

| Game KC-AE$^e, \Xi^d, A$ | Proc. KCENC($\xi^*$): $(K, M, N, H) \leftarrow \xi^*|1(K^*)$ | Proc. KCDEC($\xi^d$, C): $(K, N, H) \leftarrow \xi^d|1(K^*)$ |
|--------------------------|-------------------------------------------------|--------------------------------------------------|
| $b \leftarrow \{0, 1\}$ | $C_1 \leftarrow Enc(K, M, N, H)$ if $T[K, M, N, H] =$ undef: $T[K, M, N, H] \leftarrow \{0, 1\} C_1$ | $M_1 \leftarrow Dec(\xi^d, C, N, H)$ if $(K, C, N, H) \in CL$: return $\perp$ |
| $K^* \leftarrow A$ | $C_0 \leftarrow T[K, M, N, H]$ | $M_0 \leftarrow \perp$ |
| $b' \leftarrow A^{Gen,KCDec}$ | $CL \leftarrow CL : (K, C_0, N, H)$ | return $C_b$ |
| return $(b' = b)$ | | |

Figure 7: the game defining the KC-AE security of SE with key-correlated inputs and private headers and nonces (which are not revealed to the adversary). The adversary is required not to repeat nonces in its KCENC queries.
Figure 8: Game0 shows the KC-AE game with respect to $SE$ and challenge bit 1. Game1 separates the lists for $H$ and $H'$. There are no functional changes. Game2 then fully decouples $H$ and $H'$.

Overall, our reduction proceeds by choosing a $K^*$, computing the key-correlated message, nonce, and header, faithfully using this key; and implicitly setting $H(x _ {i} | 1(K^*) | N)$ to keys $K_i$ in the MUAE game. For this analysis, we need that the keys $K_i$ remain hidden from the adversary’s point of view, which we argue is the case as long the queried functions are key-unpredictable. We first give a definition formalizing the unpredictability of correlated keys before formally stating and proving the security guarantees of $SE$.

**Theorem 3** (KCA security of HwN). Let $SE$ be an AE-secure AEAD scheme. Let $\Xi^e$ and $\Xi^d$ be two CDF sets. Then for any $(\Xi^e, \Xi^d)$-KC-AE adversary $A$ against the scheme $SE$ obtained by applying the HwN transform to $SE$ that makes $q_e$ random oracle queries, $q_d$ encryption queries and $q_d$ decryption queries, there are adversaries $B$, $C_1$ and $C_2$ such that

$$\text{Adv}_{\text{KC-AE}}^{\Xi^e, \Xi^d, A}(SE) \leq 2 \cdot \text{Adv}_{\text{SE}}^{\mu(a), B}(SE) + 2 \cdot \text{Adv}_{\text{SE}}^{\mu(a), C_1}(SE) + 2 \cdot \text{Adv}_{\text{SE}}^{\mu(a), C_2}(SE),$$

where adversary $B$ makes at most $q_e + q_d$ queries to $\text{INIT}$, at most $q_e$ queries to encryption, and at most $q_d$ queries to decryption, adversary $C_1$ outputs two lists of sizes at most $q_e$ and $q_d$, and adversary $C_2$ outputs two lists of sizes at most $q_d$ and $q$.

**Proof.** The proof proceeds via a sequence of games as shown in Figs. 8 and 9. The three games in Fig. 8 run according to the KC-AE game with the HwN transform applied, and
with \( b = 1 \). When \( b = 1 \), all oracle queries to \( KCEnc \) return encryptions of key-derived inputs under related keys. Queries to \( KCDec \) return the plaintext, or the error symbol \( \perp \) if the ciphertext is deemed illegitimate. The three games in Fig. 9 run according to the KC-AE game with the HwN transform applied, and with \( b = 0 \). In this scenario, all oracle queries to \( KCEnc \) return random ciphertexts, and \( KCDec \) always returns the error symbol \( \perp \). All games, adversaries, and transitions are outlined below.

**Game\(_0\)** describes the KC-AE game with respect to the transformed scheme and \( b = 1 \).

**Game\(_1\)** is identical to **Game\(_0\)** with one conceptual change: it proceeds by separating the book-keeping list for \( H \) into cases when \( H \) is accessed within \( KCEnc \) and \( KCDec \) and when directly accessed by the adversary. More precisely, in **Game\(_1\)**, records of adversary \( A \)'s queries to the random oracle \( H \) will be recorded in a list \( L \), whereas the oracle queries from \( KCEnc \) and \( KCDec \) will be recorded in a list \( L' \) by oracle denoted \( H' \). A bad flag, \( \text{Bad} \), is set if \( H' \) is queried on a key/nonce pair that has already been queried by \( A \) to \( H \). In this game, \( H \) and \( H' \) jointly implement \( H \) in **Game\(_0\)** and event \( \text{Bad} \) prepares us to fully decouple \( H \) and \( H' \) later:

\[
\Pr[\text{Game}\(_0\)] = \Pr[\text{Game}\(_1\)] .
\]

**Game\(_2\)** is identical to **Game\(_1\)** up to the point that \( \text{Bad} \) is set. If this event occurs within \( H' \), records are no longer returned. By the fundamental lemma of game playing [BR06], we bound the difference between **Game\(_2\)** and **Game\(_1\)** as

\[
\Pr[\text{Game}\(_1\)] - \Pr[\text{Game}\(_2\)] \leq \Pr[\text{Game}\(_2\) \text{ sets } \text{Bad}] .
\]

Event \( \text{Bad} \) corresponds to a non-empty intersection between the two lists \( L \) and \( L' \). Below, we will bound \( \Pr[\text{Bad}] \) in **Game\(_1\)** by showing that (1) this probability is close to the probability of setting \( \text{Bad} \) in **Game\(_2\)** down to the MUAE security of \( SE \); and (2) the probability of setting \( \text{Bad} \) in **Game\(_2\)** is small down to the key unpredictability of \( \Xi^e \) and \( \Xi^d \).

**Game\(_3\)** switches the challenge bit to \( b = 0 \). We show below that any adversarial advantage in distinguishing **Game\(_2\)** and **Game\(_3\)** can be converted, via an adversary \( B_1 \), against the MUAE game:

\[
\Pr[\text{Game}\(_2\)] - \Pr[\text{Game}\(_3\)] = \text{Adv}_{SE}^{\text{mu-ae}}(B_1) .
\]

**Game\(_4\)** reverses the change in **Game\(_2\)** and merges random oracles \( H \) and \( H' \). We have that

\[
\Pr[\text{Game}\(_3\)] - \Pr[\text{Game}\(_4\)] \leq \Pr[\text{Game}\(_3\) \text{ sets } \text{Bad}] ,
\]

where we choose **Game\(_3\)** to analyze the probability of \( \text{Bad} \). Below we construct adversaries to show that this probability is bounded above by the key-unpredictability advantages against \( \Xi^e \) and \( \Xi^d \).

**Game\(_5\)** is identical to **Game\(_4\)** except for a reversal of the conceptual change seen between **Game\(_0\)** and **Game\(_1\)**. The random oracles are completely re-coupled, and all queries by \( A \), \( KCEnc \) and \( KCDec \) are written to a single list \( L \). There are no procedural changes in this game:

\[
\Pr[\text{Game}\(_4\)] = \Pr[\text{Game}\(_5\)] .
\]

**The reductions.** We now give the details of the bounds for the three transitional changes above.
Figure 9: $\text{Game}_3$ changes the challenge bit to 0. $\text{Game}_4$ merges $H$ and $H'$. $\text{Game}_5$ is functionally identical to $\text{Game}_4$ with merged $H$ and $H'$. $\text{Game}_6$ is the KC-AE game with respect to $\text{SE}$ and challenge bit 0.

Adversary $B_1$ plays the MUAE game and is built based on an adversary $A$ that attempts to distinguish $\text{Game}_2$ and $\text{Game}_3$ as follows. Adversary $B_1$ initializes $q$ keys and chooses a $K^*$. It then simulates the encryption and decryption oracles by explicitly encoding the actual key, and also message when answering encryption, used using $K^*$. In doing so, $B_1$ implicitly programs $H(\xi_6(K^*), N_i)$ to $K_i$, one of the $q$ keys initialized by $B_1$'s challenger. Since nonces are not reused, and each $K_i$ is chosen randomly, this will constitute a perfect simulation of the random oracle $H'$. When $b = 0$, this is a perfect simulation of $\text{Game}_4$ environment for $A$ by $B_1$. However when $b = 1$, although simulation encryption is good, care must be taken in decryption as decryption of $(K, C, N, H)$ is not allowed in the KC-AE game if $(K, C, N, H)$ was added to list CL. Adversary $B_1$ thus keeps track of these values and answers with $\perp$ when such a query arises. Note that $B_1$ once again uses its knowledge $K^*$ and further that this modification does not affect the simulation when $b = 0$. Adversary $B_1$ continues in this way and returns the bit that $A$ outputs. Hence,

$$\text{Adv}^{\text{mu-ae}}_{\text{SE}}(B_1) = \Pr[b' = 1 | b = 1] - \Pr[b' = 1 | b = 0] = \Pr[\text{Game}_4^A] - \Pr[\text{Game}_3^A].$$

Adversaries $C_i$ for $i = 1, 2$ are used to show that the probability of $\text{Bad}$ is small by the key-unpredictability of the queried functions. Adversaries $C_i$ for $i = 1, 2$ run $A$ in
Adversary $B_1^{\text{Init, Enc, Dec}}$:
\[
i \leftarrow 0; \ K' \leftarrow K.
\]
Call $B_2$, \text{Init}, $q_t + q_d$ times
\[
b' \leftarrow A^{\text{KC-Enc, KC-Dec}, H}
\]
return $b'$

KCEnc($\xi'$):
\[
[K, M, N, H] \leftarrow \xi'(K^*)
\]
if $T[K, N] = \text{undef}$:
\[
i \leftarrow i + 1; T[K, N] \leftarrow i
\]
$C \leftarrow B_1, \text{Enc}(T[K, N], M, N, H)$
\[
C \leftarrow C \cup (K, C, N, H)
\]
return $C$

KCDec($\xi'$, $C$):
\[
[K, N, H] \leftarrow \xi(K^*)
\]
if $T[K, N] = \text{undef}$:
\[
i \leftarrow i + 1; T[K, N] \leftarrow i
\]
if $(K, C, N, H) \in CL$: return $\bot$
\[
M \leftarrow B_2, \text{Dec}(T[K, N], C, N, H)
\]
return $M$

$H(K|N)$:
\[
\text{if } (K|N, K_h) \in L: \text{return } K_h
\]
else $K_h \leftarrow K$
\[
L \leftarrow L \cup (K|N, K_h)
\]
return $K_h$

Adversary $B_2^{\text{Init, Enc, Dec}}$:
\[
i \leftarrow 0; \ K' \leftarrow K.
\]
Call $B_2$, \text{Init}, $q_t$ times
\[
b' \leftarrow A^{\text{KC-Enc, KC-Dec}, H}
\]
for $(\xi) \in L_1 \land (K, N) \in L'$
\[
\text{if } \xi_1(K^*) = K \land \xi_2(K^*) = N:
\]
\[
\text{return } 1
\]
\[
\text{return } 0
\]

Figure 10: On the left is the description of adversary $B_1$ simulating the challenger of $A$ in the KC-AE game whilst interacting in the MUAE environment. In the middle is the description of adversaries $C_i$ simulating the oracle requests by $A$ in the KC-AE game when $b = 0$ and whilst running in the key unpredictability environment. On the right is the code for an adversary $B_2$ simulating $A$ in the KC-AE game whilst interacting in the MUAE environment.

$\text{Game}_3$ by simply simulating its encryption and decryption oracles with $\$ \text{ and } \bot$ respectively. They also lazily sample the random oracles $H$ for $A$. Adversary $C_1$ keeps track of all queried functions to KCEnc and adversary $C_2$ keeps track of all queried functions to KCDec. Both adversaries also keep track of all queries to $H$. When $A$ terminates, $C_i$ simply outputs the list of all queried functions that it keeps track of and $H$ queries its two lists of guesses in the multi-shot key-unpredictability game. (See Fig. 10 (middle) for the details.) Whenever flag Bad is triggered, either $C_1$ or $C_2$ wins the key-unpredictability game against $\Xi^c$ or $\Xi^d$ respectively:
\[
\Pr[\text{Game}_3 \text{ sets Bad}] \leq \text{Adv}_{\Xi^c}^{m\text{-kup}}(C_1) + \text{Adv}_{\Xi^d}^{m\text{-kup}}(C_2).
\]

**Adversary $B_2$** is used to bound the difference between the probabilities of setting Bad in games $\text{Game}_3$ and $\text{Game}_4$. This is done via another reduction to the MUAE game which attempts to decide the challenge bit by observing if Bad is set. Algorithm $B_2$
runs $A$ similarly to algorithm $B_1$. It however keeps track of the queried functions to KCENC and KCDEc as well as $H$. It uses its knowledge of $K^*$ to verify if $\text{Bad}$ has occurred. When the MUAE challenge bit of $B_2$ is 1, algorithm $B_2$ runs $A$ in an environment identical to that of $\text{Game}_2$; hence the probability of $B_2$ outputting 0 is exactly that of $\text{Bad}$ in $\text{Game}_2$. Similarly, the probability of $B_2$ outputting 1 is exactly that of $\text{Bad}$ occurring in $\text{Game}_3$. Hence

$$\text{Adv}_{\text{mu-ae}}^{\text{SE}}(B_2) = \Pr[\text{Game}_2^{A} \text{ sets } \text{Bad}] - \Pr[\text{Game}_3^{A} \text{ sets } \text{Bad}] .$$

The theorem follows by putting the bounds established above together, and noting that

$$\text{Adv}_{\text{kc-ae}}^{\text{SE}}(\Xi^e, \Xi^d, A) = \Pr[\text{Game}_0^{A}] - \Pr[\text{Game}_5^{A}] .$$

In the theorem statement, we choose $B$ to be $B_1$ or $B_2$ having the bigger advantage. 

**Remark.** The above theorem can be extended to a misuse-resilient and key-correlated setting, whereby nonces may be repeated at encryption for a construction similar to HwN that also hashes the header information. A similar proof strategy can be used to analyse its KC-MRAE security at the expense of introducing three extra conditions: (1) the base scheme is MRAE secure; (2) the sets $\Xi^e$ and $\Xi^d$ are claw-free; and (3) $(\Xi^e, \Xi^d)$ is xkd so that illegal decryption queries can be detected.

**Acknowledgments**

We thank the reviewers of the IACR Transactions on Symmetric Cryptology (ToSC) for their detailed comments, which helped to improve the quality of the paper. We also thank Martijn Stam for his participation in the early stages of this work.

Farshim was supported by the European Research Council under the European Community’s Seventh Framework Programme (FP7/2007-2013 Grant Agreement no. 339563 - CryptoCloud). Fuchsbauer was supported by the MSR–Inria Joint Centre and the French ANR project $E$TrEC (ANR-16-CE39-0002).

**References**


A Relation between KCA, RKA, and KDM

RKA and KDM security. Fig. 11 gives the definitions of RKA security [BK03] and KDM security [FKV17] for blockciphers. The advantage functions are defined in the standard way (as for KCA, Equation (1) in Sec. 3). The CPA versions are obtained naturally by disallowing decryption queries.

We now prove that for RKA-only (resp. KDM-only) claw-free sets KCA and RKA (resp. KDM) security are equivalent. For convenience we restate Fig. 1 as Fig. 1. Let $A$ be a blockcipher. Let $\Xi^e = \Xi^d = \Phi \times \Gamma$. Then for every adversary $B$ there exists an adversary $A$ such that

$$\text{Adv}_{BC}^{\text{RKA}}(\Phi, B) \leq \text{Adv}_{BC}^{\text{KCA}}(\Xi^e, \Xi^d, A) + q^2/2^n,$$

where $n$ is the block length and $q$ is the maximum number of queries $B$ makes. Moreover, for every adversary $A$ there exist adversaries $B$ and $B^d$ such that

$$\text{Adv}_{BC}^{\text{KCA}}(\Xi^e, \Xi^d, A) \leq q^2 \cdot (\text{Adv}_{B^d}^{\text{KDM}}(B^d) + 1/2^n) + \text{Adv}_{BC}^{\text{KCA}}(\Phi, B),$$

where $q$ is the maximum number of queries adversary $A$ makes.

Recall that $(\Xi^e, \Xi^d)$ is RKA-only if $\Xi^e = \Xi^d = \Phi \times \Gamma$ for an RSA set $\Phi$ of functions mapping keys to keys (where $\Gamma$ is the set of constant functions over $M$). The difference
between games RK and KC is that in the former there are no lists ML and CL, thus all queries are allowed. When showing KC⇒RK, the reduction must therefore answer all queries, while its KC challenger might reply with ⊥. The reduction will therefore keep local lists CL′ and ML′, which simulate the challenger’s blacklists.

The KC challenger, when queried KCEnc(ξe = (φe, M)) stores pairs (K = φe(K∗), C = E(K, M)) in its list CL. The reduction does not know the values K and will store pairs (C, M) in a list CL′. In the KC game, queries KCDec(ξd = (φd, C)) with (φd(K∗), C) ∈ CL are answered with ⊥, whereas in the RK game, RKDec(φd, C) always returns D(φd(K∗), C). To answer such queries, the reduction looks for an entry (C, M′) for some M′ in its list CL′. If (1) there is only one such entry, it returns M′, which perfectly simulates RKDec: if (C, M′) ∈ CL′ and KCDec(ξd = (φd, C)) returned ⊥ then (φd(K∗), C) ∈ CL and if there is only one entry with C, we must have φd(K∗) = φd(K∗). If (2) there are several entries (C, M0), (C, M1) ∈ CL′, the reduction aborts the simulation and returns 1 (that is, it guesses that it is interacting with the actual blockcipher). As we show in the lemma below, we can bound the probability of this type of collision in CL′ in case the reduction is interacting with an ideal cipher, and therefore the probability of aborting and returning a wrong guess. The list ML and queries to the KCEnc oracle are dealt with analogously.

**Lemma 1.** Consider game KC-CCA(0, C) with b fixed to 0. Then the probability that there are two entries (K0, C), (K1, C) ∈ CL with iD(K0, C) ̸= iD(K1, C) is at most 2q2/2n, where q is the number of queries made by A. The same holds for the probability of two entries (K0, M), (K1, M) ∈ ML with iE(K0, M) ̸= iE(K1, M).

**Proof.** Let us consider CL (the case ML is completely analogous). In order for a pair (K, C) to be added to CL, the adversary must make a query KCEnc(ξe = (φe, M)) such that K = φe(K∗) and C = iE(K, M).

Since (iE, iD) is ideal, the probability that after q queries the adversary finds K0, K1, M0 ̸= M1 such that iE(K0, M0) = iE(K1, M1) =: C, without having queried both iD(K0, C) and iD(K1, C) before, is at most 2q2/2n. On the other hand, when the adversary queries both KCDec(φ0, C) and KCDec(φ1, C) with φi(K∗) = K, then (K0, M0) and (K1, M1), with M1 := iD(K1, C), are both added to ML. But then any call to KCEnc(ξe = (φ′, M′)) with φ′(K∗) = K1 will be answered with ⊥, meaning that (K1, C) will not be added to CL. So the only way for the event to happen is that the adversary guesses K0, K1, M0 ̸= M1 which map to the same C.

**KC ⇒ RK.** We now prove the first statement of Proposition 2. Consider an adversary B against RK; we construct a reduction A against KC, which maintains local lists CL′ and ML′, allowing it to simulate B’s oracles even when its own oracles return ⊥.

Whenever adversary B queries RKEnc on (φ, M), reduction A queries its own oracle KCEnc on ξe: K ↦ (φ(K), M). If its answer is ⊥ then there must be an entry (M, C′) in ML′. If there is more than one such entry, A stops and returns 1; otherwise it returns C′. If A’s KCEnc oracle returned a value C′ ̸= ⊥, B returns C and adds (C, M) to its local list CL′.

Analogously, if B queries RKDec on (φ, C), reduction A queries its own oracle KCDec on ξd: K ↦ (φ(K), C). If its answer is ⊥ then there must be an entry (C, M′) in CL′. If there is more than one such entry, A stops and returns 1; otherwise it returns M′. If A’s KCDec oracle returned a value M′ ̸= ⊥, B returns M and adds (M, C) to its list ML′.

If A has not stopped the simulation, it forwards B’s output b′.

First note that to any entry (K, C) ∈ CL of A’s challenger corresponds an entry (C, D(K, M)) in A’s local list CL′ (and similarly for ML and ML′). From Lemma 1 we thus have that in case b = 0, the probability that there are two entries (C, M0), (C, M1) with M0 ̸= M1 in CL′ is at most q2/2n := ε. Thus, the probability that A stops the simulation
in case $b = 0$ is bounded by $\epsilon$. In case $b = 1$, if $A$ stops the simulation, it outputs the correct bit. More formally, let $E_i$ denote the event that $A$ stops the simulation when its challenger’s bit $b = i$ and let $KC_i$ denote game $KC-CCA^{\Xi', \Xi^d, \phi}$ with bit $b$ fixed to $i$ and similarly for $RK_i$. Then we have:

$$
\text{Adv}^{rk-cca}_{BC}(\Phi, B) + 1 = \Pr[RK_0[E_0] \Pr[E_0] + \Pr[RK_0[\neg E_0] \Pr[\neg E_0] \\
+ \Pr[RK_1[E_1] \Pr[E_1] + \Pr[RK_1[\neg E_1] \Pr[\neg E_1] \\
= \Pr[RK_0[E_0] \Pr[E_0] + \Pr[KC_0[\neg E_0] \Pr[\neg E_0] \\
+ \Pr[RK_1[E_1] \Pr[E_1] + \Pr[KC_1[\neg E_1] \Pr[\neg E_1] \\
\leq 1 \cdot \epsilon + \Pr[KC_0[\neg E_0] \Pr[\neg E_0] + 1 \cdot \Pr[E_1] + \Pr[KC_1[\neg E_1] \Pr[\neg E_1] \\
= \epsilon + \Pr[KC_0[E_0] \Pr[E_0] + \Pr[KC_0[\neg E_0] \Pr[\neg E_0] \\
+ \Pr[KC_1[E_1] \Pr[E_1] + \Pr[KC_1[\neg E_1] \Pr[\neg E_1] \\
= \epsilon + \text{Adv}^{kc-cca}_{BC}(\Xi^c, \Xi^d, A) + 1 ,
$$

where for the second equality we used that games $RK$ and $KC$ have the same output conditioned on $\neg E$; the inequality uses $\Pr[E_0] \leq \epsilon$ by Lemma 1; and the second to last equality follows from $B$’s behavior in case of $E$ happening, thus $\Pr[KC_0[E_0] = 0$ and $\Pr[KC_1[E_1] = 1$. This concludes this direction of the proof.

The above simulation strategy for does not work for the converse implication $RK \Rightarrow KC$, as best illustrated by the following example: Consider a blockcipher for which $\Rightarrow$ there exist $K \neq K' \in \mathcal{K}$ and $M \in \mathcal{M}$ with $E(K, M) = E(K', M) = C$; now consider a $KC$ adversary $A$ that queries $KCEnc(\phi, M)$ with $\phi(K^*) = K$; thus $(K, C) \in CL$. Later $A$ queries $KCDec(\phi', C)$ and $KCDec(\phi'', C)$ where $\phi'(K^*) = K'$ and $\phi''(K^*) = K'$. Reduction $B$ can forward these queries to its RK oracle and will get $M$ in both cases. But $KCDec(\phi', C)$ should be answered with $M$ (since $(K', C) \notin CL$, whereas $KCDec(\phi'', C)$ should be answered with $\bot$. Thus, storing $(C, M)$ in a local list $CL'$ doesn’t help, since both queries would make $B$ look at this entry, but should result in different actions.

The reduction can thus not perfectly simulate the $KC$ experiment; although this situation is very unlikely to occur for the ideal cipher, the reduction cannot abort and return 1 (as in the above proof), since the event that the adversary found a tuple $(K, K', M, C)$ that would be hard to find for the ideal cipher is not detectable by the reduction (in contrast to the proof for $KC \Rightarrow RK$).

The proof will now rely on claw-freeness of the set $\Phi$. If it was perfectly claw-free, then the reduction could perfectly simulate entries $(K, C) \in CL$ by entries $(\phi, C) \in CL'$ with $\phi(K^*) = K$. But if claw-freeness is only computational then the adversary could find $\phi$, $\phi'$ with $\phi(K^*) = \phi'(K^*)$ for the key $K^*$ chosen by the game, but which would not be a claw for a random key and thus cannot be output as a solution in the claw-freeness game.

However, we can show that until the adversary finds a claw, the ideal game can be simulated without having chosen any key $K^*$ at all. If the adversary thus finds a claw, it must be one for a random key with non-negligible probability as well. We start with proving this last observation in the following lemma.

**Lemma 2.** Consider game $RK-CCA^{\Phi, B}$ with $b$ fixed to 0 and define event $E$ as follows: the adversary $B$ makes two $RKEnc$ queries $(\phi, M)$ and $(\phi', M)$, with $\phi \neq \phi'$, both answered by the same $C$; or it makes two $RKDec$ queries $(\phi, C)$, $(\phi', C)$, with $\phi \neq \phi'$, both answered by the same $M$; or it makes two queries $RKEnc(\phi, M)$, answered by $C$ and $RKDec(\phi', C)$, answered by $M$, with $\phi \neq \phi'$.

Then there exists an adversary $A^{cl}$ such that $\Pr[E] \leq q^2 \cdot (\text{Adv}_2(\Phi, A^{cl}) + 1/2^n) =: \epsilon_2$, where $q$ is the number of oracle queries $B$ makes in game $RK-CCA$.

**Proof.** The crucial observation is that game $RK-CCA^{\Phi, B}$ when $b$ is fixed to 0 can be simulated without knowing choosing $K^*$ up to the event $E$ happening, that is, the point
where the adversary produces the first claw. Let \((\phi_i)_{i=1}^q\) denote the list of all functions queried to \(\text{RKEnc}\) or \(\text{RKDec}\). We construct an adversary \(A^{\sf{cf}}\) against claw-freeness as follows: it first guesses \(i_0, i_1 \in [1, q]\) and then simulates the game for \(B\) until the \(i_1\)-th query as follows:

\(A^{\sf{cf}}\) keeps a list \(L\) with entries of the form \((\phi, M, C)\). Whenever \(B\) queries \(\text{RKEnc}(\phi, M)\), it checks whether there is some entry \((\phi, M, C') \in L\) and if so, returns \(C'\). Else it samples \(C \leftarrow \mathcal{M}\), adds \((\phi, M, C)\) to \(L\) and returns \(C\). Whenever \(B\) queries \(\text{RKDec}(\phi, C)\), it checks whether there is some entry \((\phi, M', C) \in L\) and if so, returns \(M'\). Else it samples \(M \leftarrow \mathcal{M}\), adds \((\phi, M, C)\) to \(L\) and returns \(M\). When \(B\) makes its \(i_1\)-th query, \(A^{\sf{cf}}\) stops and returns \((\phi_{i_0}, \phi_{i_1})\).

Assume event \(E\) occurred and \(A^{\sf{cf}}\) guessed correctly that the \(i_1\)-th query was the first query after which \(E\) occurred and the \(i_0\)-th query was the one it collided with. Consider \(A^{\sf{cf}}\)'s challenger, which chooses \(K^* \leftarrow \mathcal{K}\) and checks whether \(\phi_{i_0}(K^*) = \phi_{i_1}(K^*)\). If so, then \(A^{\sf{cf}}\) won the claw-freeness game. Else the probability that \(A^{\sf{cf}}\) would have answered the \(i_1\)-th query so that \(E\) still occurred is \(1/2^n\).

\(\text{RK} \Rightarrow \text{KC}\). This direction relies on the claw-freeness of the set \(\Phi\). Let \(A\) be a KC adversary; we construct a reduction \(\mathcal{B}\) against \(\text{RK}\) that maintains two local lists \(L'\) and \(\mathcal{CL}'\) and aborts whenever event \(E\), defined in Lemma 2, occurs.

Whenever \(A\) queries \(\text{KCEnc}\) for \(\xi^*: K \mapsto (\phi(K), M)\), reduction \(\mathcal{B}\) queries \(\text{RKEnc}(\phi, M)\) to receive \(C\). It first checks whether there has been a “colliding” \(\text{RKEnc}\) query: if there is an entry \((\phi', C, M)\) in its list \(\mathcal{CL}'\) with \(\phi' \neq \phi\) then \(\mathcal{B}\) aborts and returns 1.

To answer the query \(\text{KCEnc}(\phi, M)\), an actual KC challenger would check whether \((\phi(K^*), M) \in L\). Instead, \(B\) checks its local list \(L'\): if \((\phi, M, C) \in L'\), it returns \(\bot\); if there is an entry \((\phi' \neq \phi, M, C)\) then \(B\) aborts the simulation and returns 1; if there is no entry \((\cdot, M, C) \in L'\), it adds \((\phi, M, C)\) to its local list \(\mathcal{CL}'\) and returns \(C\).

\(\text{KCDec}\) queries for \(\xi^d: K \mapsto (\phi(K), C)\) are dealt with analogously: \(B\) first queries \(\text{RKDec}(\phi, C)\) to receive \(M\). If there is an entry \((\phi', M, C) \in L'\) with \(\phi' \neq \phi\) then \(B\) aborts and returns 1. To answer \(\text{KCDec}(\phi, C)\), an actual KC challenger would now check whether \((\phi(K^*), C) \in \mathcal{CL}\). Instead, \(B\) checks its local list \(L'\): if \((\phi, C, M) \in L'\) it returns \(\bot\); if there is an entry \((\phi' \neq \phi, C, M)\) then \(B\) aborts the simulation and returns 1; if there is no entry \((\cdot, M, C) \in L'\), it adds \((\phi, M, C)\) to its local list \(L'\) and returns \(M\).

We show that the simulation is perfect if \(B\) does not abort; consider a query \(\text{KCDec}(\phi, C)\):

1. If \((\phi, C, M) \in \mathcal{L}'\) then there must have been a query \(\text{KCEnc}(\phi, M)\) with \(C = E(\phi(K^*), M)\); thus a KC challenger would have added \((\phi(K^*), C)\) to \(\mathcal{CL}\) and the above \(\text{KCDec}\) query would have been answered by \(\bot\).
2. If there is no entry \((\cdot, C, M) \in \mathcal{L}'\) then \((\phi(K^*), C)\) cannot be in \(\mathcal{CL}\), since whenever it is added to \(\mathcal{CL}\), an entry \((\cdot, C, D(\phi(K^*), C)\) would have been added to \(\mathcal{L}'\). Perfect simulation of \(\text{KCEnc}\) queries is argued completely analogously.

On the other hand, we have that \(B\) aborts precisely when \(E\), as defined in Lemma 2, occurs. With \(\epsilon\) as defined in the latter, we thus have

\[
\text{Adv}_{\text{KC-CCA}}^{\text{BC}}(\Xi^e, Ξ^d, A) + 1 = \Pr[\text{KCDec}[E_0]\mid E_0] + \Pr[\text{KCDec}[\neg E_0]\mid \neg E_0] + \Pr[\text{RKDec}[E_0]\mid \neg E_0] + \Pr[\text{RKDec}[-E_0]\mid -E_0] + \Pr[\text{CKDec}[E_1]\mid -E_1] + \Pr[\text{RKDec}[E_1]\mid -E_1] + \Pr[\text{CKDec}[\neg E_1]\mid \neg E_1] + \Pr[\text{RKDec}[\neg E_1]\mid \neg E_1] \\
\leq 1 + \epsilon + \Pr[\text{RKDec}[E_0]\mid \neg E_0] + \Pr[\text{RKDec}[-E_0]\mid -E_0] + \Pr[\text{CKDec}[E_1]\mid -E_1] + \Pr[\text{RKDec}[E_1]\mid -E_1] + \Pr[\text{CKDec}[\neg E_1]\mid \neg E_1] + \Pr[\text{RKDec}[\neg E_1]\mid \neg E_1] \\
= \epsilon + \text{Adv}_{\text{BC}}^{\text{KC-CCA}}(\Phi, B) + 1 ,
\]
where for the second equality we used that games RK and KC have the same output conditioned on \( \neg E \); the inequality uses \( \Pr[E_0] \leq \epsilon \) by Lemma 2; and the second to last equality follows from \( B \)'s behavior in case of \( E \) happening, thus \( \Pr[RK_0|E_0] = 0 \) and \( \Pr[RK_1|E_1] = 1 \). This concludes this direction of the proof. \( \square \)

### A.2 KC(KDM-only) \( \iff \) KDM

**Proposition 3.** Let \( BC \) be a blockciper. Let \( \Xi^e \) and \( \Xi^d \) be two KDM-only CDF sets with \( \Xi^e = \{id\} \times \Psi \) and \( \Xi^d = \{id\} \times \Gamma \). Then for every adversary \( A \) there exists an adversary \( B \) such that

\[
Adv_{BC}^{kc-cca}(\Xi^e, \Xi^d, A) = Adv_{BC}^{kdm-cca}(\Psi, B),
\]

and for every adversary \( B \) there exist adversaries \( A \) and \( A^d \) such that

\[
Adv_{BC}^{kdm-cca}(\Psi, B) \leq q^2 \cdot Adv_{\Psi}^{cf}(A^d) + Adv_{BC}^{kc-cca}(\Xi^e, \Xi^d, A),
\]

where \( q \) is the maximum number of queries adversary \( B \) makes.

We show that our definition is equivalent to the KDM definition from [FKV17] for claw-free sets \( \Psi \). Recall that \( (\Xi^e, \Xi^d) \) is KDM-only if \( \Xi^e = \{id\} \times \Psi \) and \( \Xi^d = \{id\} \times \Gamma \), where \( \Psi \) is a KDM set of functions mapping keys to messages and \( \Gamma \) is the set of constant functions \( K \mapsto M \) for all \( M \in M \); we assume \( \Gamma \subseteq \Psi \).

We first observe that games KC and KDM are similar in that every entry \( C \) in CL in game KDM corresponds to \( (K^*, C) \) in CL in KC. The crucial difference is that in KDM there is no list ML, thus all KDMEnc queries are allowed, whereas KC disallows queries in ML. KDM implying KC can be shown because the reduction can perfectly simulate the list ML.

The converse (KC \( \Rightarrow \) KDM) is trickier to show and it relies on claw-freeness of the set \( \Psi \). To illustrate this, consider the following attack against KDM, which does not work against KC: for two random \( C_0, C_1 \in M \), the adversary queries Dec\((C_0)\) and Dec\((C_1)\) to receive \( M_0 \) and \( M_1 \). It then queries KDMEnc\((\psi_{M_0,M_1})\) with \( \psi : K^* \mapsto M_{K^*_1} \), where \( K^*_1 \) is the first bit of \( K^* \). When receiving \( C_i \) as a reply, the adversary learned that \( i \) is the first bit of \( K^* \) and it can use the same strategy to learn all other bits of \( K^* \). Note that in the KC game, the two decryption queries add \( (K^*, M_0) \) and \( (K^*, M_1) \) to ML and consequently KCEnc\((\xi^e := id \times \psi_{M_0,M_1})\) is answered with \( \perp \).

**KDM \( \Rightarrow \) KC.** To show that KDM implies KC, consider a KC adversary \( A \) and let us construct a reduction \( B \) against KDM. \( B \) basically simulates the KC oracles for \( A \) by forwarding \( A \)'s queries to its own KDM oracles. To simulate game KC correctly, \( B \) maintains its own list ML'. Whenever \( A \) queries KCDec\((\xi^d)\) for \( \xi^d \in \Xi^d \), we have \( \xi_2: K \mapsto (K, C) \) for some \( C \in M \). Reduction \( B \) queries \( C \) to Dec to obtain \( M \), which it returns to \( A \); \( B \) also appends \( (M, C) \) to its list ML' (whereas an actual KC challenger would have appended \( (K^*, M) \) to its list ML).

Whenever \( A \) queries KCEnc\((\xi_1)\) with \( \xi_1: K \mapsto (K, \psi(K)) \) for some \( \psi \in \Psi \), the reduction queries KDMEnc\((\psi)\) to obtain \( C \). If for some \( M' \) there is an entry \( (M', C) \in ML' \), \( B \) returns \( \perp \); otherwise, it returns \( C \).

To see that this correctly simulates the list ML that \( B \)'s challenger maintains, observe that \( (K^*, M) \) is in the latter iff \( (M, E(K^*, M)) \) is in \( B \)'s local list ML' and since \( E(K^*, \cdot) \) is a permutation, each \( C \) uniquely determines the corresponding \( M \), thus \( B \)'s simulation is perfect. We thus have \( Adv_{BC}^{kdm-cca}(\Xi^e, \Xi^d, A) = Adv_{BC}^{kdm-cca}(\Psi, B) \).

**KC \( \Rightarrow \) KDM.** In this direction, we assume an adversary \( B \) against KDM and construct \( A \) against KC. This time, \( A \) will not be able to perfectly simulate the game; however, \( A \) can detect when it is not able to. In this case \( A \) aborts and returns \( b' = 1 \) (it guesses that it is talking to the real blockcipher). As for in the proof for KC \( \Rightarrow \) RK, we show
that if \(b = 0\) (the ideal case), the probability of \(A\) aborting is bounded—in contrast to the previous proof, we bound it by the advantage of breaking claw-freeness.

We start by defining the following event \(E\) in game KDM-CCA\(_{BC}\): let \((M_i)\), denote the messages returned by the Dec oracle and let \((\psi_j)\) denote the queries made to the KDM\(_{\text{Enc}}\) oracle; let \(K^*\) denote the challenge key. We say \(E\) occurs if we either have (1) \(\psi_{j_0}(K^*) = \psi_{j_1}(K^*)\) for some \(j_0 \neq j_1\); or (2) \(\psi_j(K^*) = M_i\) for some \(j, i\). We show the following:

**Lemma 3.** Consider game KDM-CCA\(_{BC}\) with \(b\) fixed to 0 and let \(E\) be the event defined above. Then there exists an adversary \(A^{cf}\) such that \(\Pr[E] \leq q^2 \cdot \text{Adv}_{\psi,\psi}^d(A^{cf})\), where \(q\) is the number of oracle queries \(\mathcal{B}\) makes in game KDM-CCA.

**Proof.** First note that there exists some first query \(j_1\) that makes \(E\) occur, and some query \(j_0 < j_1\), which will be the query with which the \(j_1\)-th query “collides”. \(A^{cf}\) starts with guessing \(j_0\) and \(j_1\) and simulates the game until the \(j_1\)-th query as follows: any query \(\text{Dec}(C_i)\) is answered with a random \(M_i \leftarrow \mathcal{M}\); any query KDM\(_{\text{Enc}}(\psi)\) is answered with a random \(C_i \leftarrow \mathcal{C}\) (all repeated queries are answered consistently).

If the \(j_0\)-th query was a query \(\text{Dec}(C_i)\) answered by \(M_i\) then define \(\psi_{j_0}: K \mapsto M_i\); if it was a query KDM\(_{\text{Enc}}(\psi)\) then let \(\psi_{j_0} := \psi\); define \(\psi_{j_1}\) analogously. \(A^{cf}\) returns \((\psi_{j_0}, \psi_{j_1})\).

First note that if \(A\) guessed \(j_1\) correctly then the simulation is perfect, as for the random key \(K^*\), the same message was never queried twice (via encryption or decryption). Moreover, if \(A\) also guessed \(j_0\) correctly then \((\psi_{j_0}, \psi_{j_1})\) is indeed a collision and \(A\) wins the claw-freeness game. We thus have \(\text{Adv}_{\psi,\psi}^d(A^{cf}) \geq 1/q^2 \cdot \Pr[E]\), which proves the lemma.

We now give a reduction \(A\) against KC that simulates the KDM game to adversary \(\mathcal{B}\) by forwarding all queries to its own oracle. When \(A\) cannot answer a KDM\(_{\text{Enc}}\) query because its own oracle KC\(_{\text{Enc}}\) replies \(\perp\) it aborts the simulation. In order to use the above lemma, \(A\) will also abort when it could continue the simulation. In particular, \(A\) aborts and returns 1 when one of the two things happen:

1. \(\mathcal{B}\) makes two queries \(\psi, \psi'\) to KDM\(_{\text{Enc}}\), which when forwarded to \(A\)'s RK\(_{\text{Enc}}\) oracle return the same \(C\); or
2. \(A\) forwards a KDM\(_{\text{Enc}}\) query to its KDM\(_{\text{Enc}}\) oracle which is answered with \(\perp\).

Note that these two cases define precisely the event \(E\) above. We thus have that \(A\) aborts the simulation whenever \(E\) occurs. If \(A\) does not abort the simulation, it returns whatever \(\mathcal{B}\) outputs.

Let KC\(_i\) denote game KC-CCA\(_{\text{BC}}\)^\_d,\_e,\_d,\_d with bit \(b\) fixed to 0 and \(E_i\) the event that \(E\) occurs in this case. Define KDM\(_i\) analogously and let \(\epsilon := q^2 \cdot \text{Adv}_{\psi,\psi}^d(A^{cf})\). Then we have:

\[
\text{Adv}_{\text{KDM-CCA}}^d(\psi, \mathcal{B}) + 1 = \Pr[\text{KDM}_0|E_0] \Pr[E_0] + \Pr[\text{KDM}_0|\neg E_0] \Pr[\neg E_0] \\
+ \Pr[\text{KDM}_1|E_1] \Pr[E_1] + \Pr[\text{KDM}_1|\neg E_1] \Pr[\neg E_1] \\
= \Pr[\text{KDM}_0|E_0] \Pr[E_0] + \Pr[\text{KDM}_0|\neg E_0] \Pr[\neg E_0] \\
+ \Pr[\text{KDM}_1|E_1] \Pr[E_1] + \Pr[\text{KDM}_1|\neg E_1] \Pr[\neg E_1] \\
\leq 1 + \epsilon + \Pr[\text{KDM}_0|\neg E_0] \Pr[\neg E_0] + \Pr[\text{KDM}_1|\neg E_1] \Pr[\neg E_1] \\
= \epsilon + \Pr[\text{KDM}_0|E_0] \Pr[E_0] + \Pr[\text{KDM}_0|\neg E_0] \Pr[\neg E_0] \\
+ \Pr[\text{KDM}_1|E_1] \Pr[E_1] + \Pr[\text{KDM}_1|\neg E_1] \Pr[\neg E_1] \\
= \epsilon + \text{Adv}_{\text{KDM-CCA}}(\Xi^e, \Xi^d, \mathcal{A}) + 1,
\]

where for the second equality we used that games KDM and KC have the same output conditioned on \(\neg E\); the inequality uses \(\Pr[E_0] \leq \epsilon\) by Lemma 3; and the second to last equality follows from \(\mathcal{B}'\)'s behavior in case of \(E\) happening, thus \(\Pr[\text{KDM}_0|E_0] = 0\) and \(\Pr[\text{KDM}_1|E_1] = 1\). This concludes this direction of the proof.
Let \( \text{Game } \): 
\[
\begin{array}{l}
\text{Game}_{i, j}^A: \\
b \leftarrow \{0, 1\}; v \leftarrow 0 \\
b' \leftarrow A^{\text{Init, Enc, Dec}} \\
\text{return } (b' = b)
\end{array}
\]

\[
\begin{array}{l}
\text{Proc. } \text{Init}(): \\
v \leftarrow v + 1; K_v \leftarrow K \\
\text{return } C_b \\
\end{array}
\]

\[
\begin{array}{l}
\text{Proc. } \text{Enc}(i, M, N, H): \\
\text{if } i \leq v: \text{return } ∥ \\
C \leftarrow \text{Enc}(K_i, M, N, H) \\
\text{if } i < j : C \leftarrow \{0, 1\}^{\lceil C \rceil} \\
\text{return } C_b \\
\end{array}
\]

\[
\begin{array}{l}
\text{Proc. } \text{Dec}(i, C, N, H): \\
\text{if } i \leq v: \text{return } ∥ \\
M \leftarrow \text{Dec}(K_i, C, N, H) \\
\text{return } M \\
\end{array}
\]

Figure 13: \text{Game } \text{MUAE}_A^A \text{ defining the multi-user AE-security of a symmetric encryption scheme SE.}

Figure 14: \text{Game }_j (0 \leq j \leq n) \text{ showing MUAE game which returns random values when } i \leq j \text{ and an encryption under } K_i \text{ otherwise.}

\section{B Multi-User to Single-User Reduction for AE}

\text{Multi-user security.} \quad \text{We define the MUAE-security of } SE = (\text{Gen, Enc, Dec}) \text{ by considering the game described in Fig. 13. The MUAE advantage of an adversary } A \text{ against an MUAE game is defined by}

\[
\text{Adv}_{SE}^{\text{mu-ae}}(A) := 2 \cdot \Pr[\text{MUAE}_A^A] - 1.
\]

Note that nonces may not be repeated for each user in Enc.

\textbf{Theorem 4.} \quad \text{Let } SE \text{ be an authenticated encryption scheme. Then for any adversary } A \text{ attacking the MUAE-security of } SE \text{ while making at most } n \text{ queries to Init, there exist AE adversaries } B_1, \ldots, B_n \text{ such that}

\[
\text{Adv}_{SE}^{\text{mu-ae}}(A) \leq \sum_{i=1}^n \text{Adv}_{SE}^{\text{ae}}(B_i).
\]

\textbf{Proof.} \quad \text{The proof proceeds via a standard hybrid argument. We start by defining a sequence of games } \text{Game}_j (0 \leq j \leq n) \text{ as shown in Fig. 14. } \text{Game}_j \text{ runs similarly to the MUAE game except that if } i \text{ (the index of the key) is less than or equal to } j, \text{ then ciphertext } C \text{ is randomly generated. For } i > j, \text{ the encryption algorithm is used under key } K_i. \text{ Thus } \text{Game}_0 \text{ is identical to the MUAE game with } b = 1 \text{ and } \text{Game}_n \text{ is identical to the MUAE game with } b = 0.

\text{Let } A \text{ be any adversary in the MUAE game. To bound the advantage of } A, \text{ note that}

\[
\text{Adv}_{SE}^{\text{mu-ae}}(A) := \Pr[\text{Game}_0^A] - \Pr[\text{Game}_n^A] = \sum_{j=0}^{n-1} (\Pr[\text{Game}_j^A] - \Pr[\text{Game}_{j+1}^A]).
\]
Adversary $B_j^{\text{Enc, Dec}}$:  
\[ b' \leftarrow A^{\text{Init, Enc, Dec}} \]
return $b'$

**Init()**:  
if $i = j + 1$:  
$K_i \leftarrow \bot$
else $K_i \leftarrow K$

$$\text{ENC}(i, M, N, H);$$
if $i \leq j$:  
$C \leftarrow \{0, 1\}^{|C|}$
if $i = j + 1$:  
$M \leftarrow B_j^{\text{Enc}}(M, N, H)$
else $M \leftarrow \text{Dec}(K_i, C, N, H)$
return $C$

$$\text{DEC}(i, C, N, H);$$
if $i \leq j$:  
$M \leftarrow \bot$
if $i = j + 1$:  
$M \leftarrow \text{Dec}(C, N, H)$
else $M \leftarrow \text{Dec}(K_i, C, N, H)$
return $M$

Figure 15: Adversary $B$ in the single-user AE game based on a multi-user AE adversary $A$.

Thus, for $j = 0, \ldots, n - 1$ it remains to bound
$$\Pr[\text{Game}_j^A] - \Pr[\text{Game}_{j+1}^A].$$
To this end, we will rely on the underlying security of SE.

**Claim.** For any $j$, and for any adversary $A$, there exists a $B_j$ such that
$$\Pr[\text{Game}_j^A] - \Pr[\text{Game}_{j+1}^A] = \text{Adv}_{\text{SE}}^{\text{AE}}(B_j).$$

The key observation is that in $\text{Game}_j$, for the first $j$ keys queried, $j$ uniform strings are returned as the ciphertext and for the remaining keys, $n - j$ encryptions are returned. In $\text{Game}_{j+1}$, $j + 1$ uniform strings and $n - j - 1$ encryptions are returned. Leveraging this difference of one real encryption at the $j + 1$ position, we can construct an AE adversary $B_j$ to emulate the environment of an MUAE adversary $A$ to break the single user AE security.

Adversary $B_j$ works as shown in Fig. 15. When the challenge bit $b$ in the AE game (that $B_j$ is playing) is 1, adversary $A$’s view of the environment is identical to that in $\text{Game}_j$. When $B_j$’s challenge bit is $b = 0$, adversary $A$’s view is identical to that in $\text{Game}_{j+1}$. When $A$ outputs a bit $b'$, games $\text{Game}_j$ and $\text{Game}_{j+1}$ check if $b' = 1$. This is equivalent to outputting $b'$, which is also the bit that $B_j$ outputs. In other words,
$$\text{Adv}_{\text{SE}}^{\text{AE}}(B_j) = \Pr[b' = 1 | b = 1] - \Pr[b' = 1 | b = 0] = \Pr[\text{Game}_j^A] - \Pr[\text{Game}_{j+1}^A].$$

\[ \square \]