Hybrid Encryption in a Multi-User Setting, Revisited

Federico Giacon Eike Kiltz Bertram Poettering

September 2, 2017

Ruhr University Bochum
firstname.surname@rub.de

Abstract

This paper contributes to understanding the interplay of security notions for PKE, KEMs, and DEMs, in settings with multiple users, challenges, and instances. We start analytically by first studying (a) the tightness aspects of the standard hybrid KEM+DEM encryption paradigm, (b) the inherent weak security properties of all deterministic DEMs due to generic key-collision attacks in the multi-instance setting, and (c) the negative effect of deterministic DEMs on the security of hybrid encryption.

We then switch to the constructive side by (d) introducing the concept of an *augmented data encapsulation mechanism* (ADEM) that promises robustness against multi-instance attacks, (e) proposing a variant of hybrid encryption that uses an ADEM instead of a DEM to alleviate the problems of the standard KEM+DEM composition, and (f) constructing practical ADEMs that are secure in the multi-instance setting.

Keywords: hybrid encryption, multi-user security, tightness

1 Introduction

Hybrid encryption and its security. Public-key encryption (PKE) is typically implemented following a hybrid paradigm: To encrypt a message, first a randomized key encapsulation mechanism (KEM) is used to establish—independently of the message—a fresh session key that the receiver is able to recover using its secret key; then a deterministic data encapsulation mechanism (DEM) is used with the session key to encrypt the message. Both KEM and DEM output individual ciphertexts, and the overall PKE ciphertext is just their concatenation. Benefits obtained from deconstructing PKE into the two named components include easier implementation, deployment, and analysis. An independent reason that, in many cases, makes separating asymmetric from symmetric techniques actually necessary is that asymmetric cryptographic components can typically deal only with messages of limited length (e.g., 2048 bit messages in RSA-based systems) or of specific structure (e.g., points on an elliptic curve). The paradigm of hybrid encryption, where the message-processing components are strictly separated from the asymmetric ones, side-steps these disadvantages.

Hybrid encryption was first studied on a formal basis in [11]. (Implicitly the concept emerged much earlier, for instance in PGP email encryption.) The central result on the security of this paradigm is that combining a secure KEM with a secure DEM yields a secure PKE scheme. Various configurations of sufficient definitions of 'secure' for the three components have been proposed [11, 18, 15], with the common property that the corresponding security reductions are tight.

Multi-user security of PKE and KEMs. Classic security definitions for PKE, like IND-CPA and IND-CCA, formalize notions of confidentiality of a single message encrypted to a single user. (For public-key primitives, we identify (receiving) users with public keys.) This does not well-reflect real-world requirements where, in principle, billions of senders might use the same encryption algorithm to send, concurrently and independently of each other, related or unrelated messages to billions of receivers. Correspondingly, for adequately capturing security aspects of PKE that is deployed at large scale,

generalizations of IND-CPA/CCA have been proposed that formalize indistinguishability in the face of multiple users and multiple challenge queries [4] (the goal of the adversary is to break confidentiality of one message, not necessarily of all messages). On the one hand, fortunately, these generalized notions turn out to be equivalent to the single-user single-challenge case [4] (thus supporting the relevance of the latter). On the other hand, and unfortunately, all known proofs of this statement use reductions that are not tight, losing a factor of $n \cdot q_e$ where n is the number of users and q_e the allowed number of challenge queries per user. Of course this does not mean that PKE schemes with tightly equivalent single- and multi-user security cannot exist, and indeed [4, 17, 19, 20, 1, 14, 12] expose examples of schemes with tight reductions between the two worlds.

The situation for KEMs is effectively the same as for PKE: While the standard security definitions [11, 15] consider exclusively the single-user single-challenge case, natural multi-user multi-challenge variants have been considered and can be proven—up to a security loss with factor $n \cdot q_e$—equivalent to the standard notions.

Multi-instance security of DEMs. Besides scaled versions of security notions for PKE and KEMs, we also consider similarly generalized variants of DEM security. More specifically, we formalize a new[1] security notion for DEMs that assumes multiple independently generated instances and allows for one challenge encapsulation per instance. (For secret key primitives, we identify instances with secret keys.) The single-challenge restriction is due to the fact that overall we are interested in KEM+DEM composition and, akin to the single-instance case [11], a one-time notion for the DEM is sufficient (and, as we show, necessary) for proving security of the hybrid. Similarly as for PKE and KEMs, the multi-instance security of a DEM is closely coupled to its single-instance security; however, generically, if N is the number of instances, the corresponding reduction loses a factor of N.

A couple of works [24, 8] observe that DEMs that possess a specific technical property[2] indeed have a lower security in the multi-instance setting than in the single-instance case. This is shown via attacks that assume a number of instances that is so large that, with considerable probability, different instances use the same encapsulation key; such key collisions can be detected, and message contents can be recovered. Note that, strictly speaking, the mentioned type of attack does *not* imply that the reduction of multi-instance to single-instance security is necessarily untight, as the attacks crucially depend on the DEM key size which is a parameter that does not appear in above tightness bounds. We finally point out that the attacks described in [24, 8] are not general but target only specific DEMs. (In this paper we show that the security of *any* DEM degrades as the number of considered instances increases.)

1.1 Our Contributions

This paper contributes to understanding the interplay of security notions for PKE, KEMs, and DEMs, in settings with multiple users, challenges, and instances. We start analytically by first studying (a) the tightness aspects of the standard hybrid KEM+DEM encryption paradigm, (b) the inherent weak security properties of deterministic DEMs in the multi-instance setting, and (c) the negative effect of deterministic DEMs on the security of hybrid encryption. We then switch to the constructive side by (d) introducing the concept of an *augmented data encapsulation mechanism* (ADEM) that promises robustness against multi-instance attacks, (e) proposing a variant of hybrid encryption that uses an ADEM instead of a DEM to alleviate the problems of the standard KEM+DEM composition, and (f) constructing secure practical ADEMs. We proceed with discussing some of these results in more detail, in the order in which they appear in the paper.

Standard KEM+DEM Hybrid Encryption. In Section 3 we define syntax and security properties of PKE, KEMs, and DEMs; we also recall hybrid encryption. Besides unifying the notation of algorithms and security definitions, the main contribution of this section is to provide a new multi-instance security notion for DEMs that matches the requirements of KEM+DEM hybrid encryption in the multi-user multi-challenge setting. That is, hybrid encryption is secure, with tight reduction, if KEM and DEM are simultaneously secure (in our sense). We further show that any attack on the multi-instance security of

[1] We are not aware of prior work that explicitly develops multi-instance security models for DEMs; however, [24] (and others) discuss the multi-instance security of symmetric encryption, and [7] considers the multi-instance security of (nonce-based) AE.

[2] The cited work is not too clear about this property; loosely speaking the condition seems to be that colliding ciphertexts of the same message under random keys can be used as evidence that also the keys are colliding. One example for a DEM with this property is CBC encryption.

the DEM tightly implies an attack on the multi-user multi-challenge security of the hybrid scheme. This implication is particularly relevant in the light of the results of Section 4.

Generic Key-Collision Attacks on Deterministic DEMs. In Section 4 we study two attacks that target arbitrary (deterministic) DEMs, leveraging on the multi-instance setting and exploiting the tightness gap between single-instance and multi-instance security. Concretely, inspired by the key-collision attacks (also known as birthday-bound attacks) from [24, 8, 7], in Section 4.1 and Section 4.2 we describe two attacks against arbitrary DEMs that break indistinguishability or even recover encryption keys with success probability $N^2/|\mathcal{K}|$, where N is the number of instances and $\mathcal{K}$ is the DEM's key space. (The reason for specifying two attacks instead of just one is that deciding which one is preferable may depend on the particular DEM.) As mentioned above, in hybrid encryption these attacks carry over to the overall PKE.

What are the options to thwart the described attacks on DEMs? One way to avoid key-collision attacks in practice is of course to increase the key length of the DEM. This requires the extra burden of also changing the KEM (it has to output longer keys) and hence might not be a viable option. (Observe that leaving the KEM as-is but expanding its key to, say, double length using a PRG is *not* going to work as our generic DEM attacks would immediately kick in against that construction as well.) Another way to go would be to randomize the DEM. Drawbacks of this approach are that randomness might be a scarce resource (in particular on embedded systems, but also on desktop computers there is a price to pay for requesting randomness[3]), and that randomized schemes necessarily have longer ciphertexts than deterministic ones. In Sections 5 to 7 we explore an alternative technique to overcome key-collision attacks in hybrid encryption without requiring further randomness and without requiring changing the KEM. We describe our approach in the following.

KEM+ADEM Hybrid Encryption. In Section 5 we introduce the concept of an augmented data encapsulation mechanism (ADEM). It is a variant of a DEM that takes an additional input: the tag. The intuition is that ADEMs are safer to use for hybrid encryption than regular DEMs, in particular in the presence of session-key collisions: Even if two keys collide, security is preserved if the corresponding tags are different. Importantly, the two generic attacks on DEMs from Section 4 do not apply to ADEMs. In Section 5 we further consider *augmented hybrid encryption* which constructs PKE from a KEM and an ADEM by using the KEM ciphertext as the ADEM tag. The corresponding security reduction is tight.

Practical ADEM Constructions. Sections 6 and 7 are dedicated to the construction of practical ADEMs. The two constructions in Section 6 are based on the well-known counter mode encryption, instantiated with an ideal random function and using the tag as initial counter value. We prove tight, beyond-birthday security bounds of the form $N/|\mathcal{K}|$ for the multi-instance security of our ADEMs. That is, our constructions provably do not fall prey to key collision attacks, in particular not the ones from [24, 8] and Section 4. Unfortunately, as they are based on counter mode, the two schemes *per se* are not secure against active adversaries. This is remedied in Section 7 where we show that an *augmented message authentication code*[4] (AMAC) can be used to generically strengthen a passively-secure ADEM to become secure against active adversaries. (We define AMACs and give a tightly secure construction in the same section.)

2 Notation

If S is a finite set, $s \xleftarrow{\$} S$ denotes the operation of picking an element of S uniformly at random and assigning the result to variable s. For a randomized algorithm A we write $y \xleftarrow{\$} \mathsf{A}(x_1, x_2, \ldots)$ to denote the operation of running A with inputs $x_1, x_2, \ldots$ and assigning the output to variable y. Further, we write $[\mathsf{A}(x_1, x_2, \ldots)]$ for the set of values that A outputs with positive probability. We denote the concatenation of strings with $\|$ and the XOR of same-length strings with $\oplus$. If $a \leq b$ are natural numbers, we write $[a\,..\,b]$ for the range $\{a, \ldots, b\}$.

[3] Obtaining entropy from a modern operating system kernel involves either file access or system calls; both options are considerably more costly than, say, doing an AES computation. While some modern CPUs have built-in randomness generators, the quality of the latter is difficult to assess and relying exclusively on them thus discouraged (see https://plus.google.com/+TheodoreTso/posts/SDcoemc9V3J).

[4] The notion of an augmented MAC appeared recently in an unrelated context: An AMAC according to [3] is effectively keyed Merkle–Damgård hashing with an unkeyed output transform applied at the end. Importantly, while the notion of [3] follows the classic MAC syntax, ours does not (for having a separate tag input).

We say a sequence $v_1, \ldots, v_n$ has a (two-)collision if there are indices $1 \le i < j \le n$ such that $v_i = v_j$. More generally, the sequence has a k-collision if there exist $1 \le i_1 < \ldots < i_k \le n$ such that $v_{i_1} = \ldots = v_{i_k}$. We use predicate $\mathbf{Coll}_k[\,]$ to indicate k-collisions. For instance, $\mathbf{Coll}_2[1,2,3,2]$ evaluates to *true* and $\mathbf{Coll}_3[1,2,3,2]$ evaluates to *false*.

Let $\mathcal{L}$ be a finite set of cardinality $L = |\mathcal{L}|$. Sometimes we want to refer to the elements of $\mathcal{L}$ in an arbitrary but circular way, i.e., such that indices x and $x + L$ resolve to the same element. We do this by fixing an arbitrary bijection $[\![\cdot]\!]_L \colon \mathbb{Z}/L\mathbb{Z} \to \mathcal{L}$ and extending the domain of $[\![\cdot]\!]_L$ to the set $\mathbb{Z}$ in the natural way. This makes expressions like $[\![a+b]\!]_L$, for $a, b \in \mathbb{N}$, well-defined. We use the shortcut notation $[\![a \twoheadrightarrow l]\!]_L$ to refer to the span $\{[\![a+1]\!]_L, \ldots, [\![a+l]\!]_L\}$ of length l. In particular we have $[\![a \twoheadrightarrow 1]\!]_L = \{[\![a+1]\!]_L\}$.

Our security definitions are based on games played between a challenger and an adversary. These games are expressed using program code and terminate when the main code block executes a 'return' command; the argument of the latter is the output of the game. We write $\Pr[G \Rightarrow 1]$ or $\Pr[G \Rightarrow \mathit{true}]$ or just $\Pr[G]$ for the probability that game G terminates by running into a 'return' instruction with a value interpreted as true. Further, if E is some game-internal event, we similarly write $\Pr[E]$ for the probability this event occurs. (Note the game is implicit in this notation.)

In Appendix A we state a couple of bounds on collision probabilities that will be used throughout this paper.

3 Traditional KEM/DEM composition and its weakness

We define PKE, KEMs, and DEMs, and give security definitions that consider multi-user, multi-challenge, and multi-instance attacks. Using the techniques from [4] we show that the multi notions are equivalent to their single counterparts, up to a huge tightness loss. We show that hybrid encryption enjoys tight security also in the multi settings. We finally show how (multi-instance) attacks on the DEM can be leveraged to attacks on the PKE.

3.1 Syntax and security of PKE, KEMs, and DEMs

Public-key encryption. A public-key encryption scheme $\mathsf{PKE} = (\mathsf{P.gen}, \mathsf{P.enc}, \mathsf{P.dec})$ is a triple of algorithms together with a message space $\mathcal{M}$ and a ciphertext space $\mathcal{C}$. The randomized key-generation algorithm $\mathsf{P.gen}$ returns a pair (pk, sk) consisting of a public key and a secret key. The randomized encryption algorithm $\mathsf{P.enc}$ takes a public key pk and a message $m \in \mathcal{M}$ to produce a ciphertext $c \in \mathcal{C}$. Finally, the deterministic decryption algorithm $\mathsf{P.dec}$ takes a secret key sk and a ciphertext $c \in \mathcal{C}$, and outputs either a message $m \in \mathcal{M}$ or the special symbol $\perp \notin \mathcal{M}$ to indicate rejection. The correctness requirement is that for all $(pk, sk) \in [\mathsf{P.gen}]$, $m \in \mathcal{M}$, and $c \in [\mathsf{P.enc}(pk, m)]$, we have $\mathsf{P.dec}(sk, c) = m$.

We adapt results from [4] to our notation, giving a game-based security definition for public-key encryption that formalizes multi-user multi-challenge indistinguishability: For a scheme PKE, to any adversary A and any number of users n we associate the distinguishing advantage $\mathbf{Adv}^{\mathsf{muc\text{-}ind}}_{\mathsf{PKE},\mathsf{A},n} := |\Pr[\mathsf{MUC\text{-}IND}^0_{\mathsf{A},n}] - \Pr[\mathsf{MUC\text{-}IND}^1_{\mathsf{A},n}]|$, where the two games are specified in Figure 1. Note that if q_e resp. q_d specify upper bounds on the number of Oenc and Odec queries per user, then the single-user configurations $(n, q_e, q_d) = (1, 1, 0)$ and $(n, q_e, q_d) = (1, 1, \infty)$ correspond precisely with standard definitions of IND-CPA and IND-CCA security for PKE.

Game $\mathsf{MUC\text{-}IND}^b_{\mathsf{A},n}$	**Oracle** $\mathsf{Oenc}(j, m_0, m_1)$	**Oracle** $\mathsf{Odec}(j, c)$
00 for all $j \in [1\,..\,n]$:	05 $c \xleftarrow{\$} \mathsf{P.enc}(pk_j, m_b)$	08 if $c \in C_j$: return $\perp$
01 $\quad(pk_j, sk_j) \xleftarrow{\$} \mathsf{P.gen}()$	06 $C_j \leftarrow C_j \cup \{c\}$	09 $m \leftarrow \mathsf{P.dec}(sk_j, c)$
02 $\quad C_j \leftarrow \emptyset$	07 return c	10 return m
03 $b' \xleftarrow{\$} \mathsf{A}(pk_1, \ldots, pk_n)$		
04 return b'		

Figure 1: PKE security games $\mathsf{MUC\text{-}IND}^b_{\mathsf{A},n}$, $b \in \{0, 1\}$, modeling multi-user multi-challenge indistinguishability for n users. Adversary A can access oracles Oenc and Odec.

The following states that the multi-user multi-challenge notion is equivalent to the traditional single-user single-challenge case—up to a tightness loss that is linear in both the number of users and the

number of challenges. The proof is in [4] (and for completeness also in Appendix B.1).

Lemma 3.1 ([4]). *For any public-key encryption scheme* PKE*, any number of users* n*, and any adversary* A *that poses at most* q_e*-many* Oenc *and* q_d*-many* Odec *queries per user, there exists an adversary* B *such that* $\mathbf{Adv}^{\mathsf{muc\text{-}ind}}_{\mathsf{PKE},\mathsf{A},n} \leq n \cdot q_e \cdot \mathbf{Adv}^{\mathsf{muc\text{-}ind}}_{\mathsf{PKE},\mathsf{B},1}$*, where* B *poses at most one* Oenc *and* q_d*-many* Odec *queries. Further, the running time of* B *is at most that of* A *plus the time needed to perform* nq_e*-many* P.enc *operations and* nq_d*-many* P.dec *operations.*

KEY ENCAPSULATION. A key-encapsulation mechanism $\mathsf{KEM} = (\mathsf{K.gen}, \mathsf{K.enc}, \mathsf{K.dec})$ for a finite session-key space $\mathcal{K}$ is a triple of algorithms together with a ciphertext space $\mathcal{C}$. The randomized key-generation algorithm K.gen returns a pair (pk, sk) consisting of a public key and a secret key. The randomized encapsulation algorithm K.enc takes a public key pk to produce a session key $K \in \mathcal{K}$ and a ciphertext $c \in \mathcal{C}$. Finally, the deterministic decapsulation algorithm K.dec takes a secret key sk and a ciphertext $c \in \mathcal{C}$, and outputs either a session key $K \in \mathcal{K}$ or the special symbol $\perp \notin \mathcal{K}$ to indicate rejection. The correctness requirement is that for all $(pk, sk) \in [\mathsf{K.gen}]$ and $(K, c) \in [\mathsf{K.enc}(pk)]$ we have $\mathsf{K.dec}(sk, c) = K$.

Like for PKE schemes we give a security definition for KEMs that formalizes multi-user multi-challenge indistinguishability: For a scheme KEM, to any adversary A and any number of users n we associate the distinguishing advantage $\mathbf{Adv}^{\mathsf{muc\text{-}ind}}_{\mathsf{KEM},\mathsf{A},n} := |\Pr[\mathsf{MUC\text{-}IND}^0_{\mathsf{A},n}] - \Pr[\mathsf{MUC\text{-}IND}^1_{\mathsf{A},n}]|$, where the two games are specified in Figure 2. Note that if q_e resp. q_d specify upper bounds on the number of Oenc and Odec queries per user, then the single-user configurations $(n, q_e, q_d) = (1, 1, 0)$ and $(n, q_e, q_d) = (1, 1, \infty)$ correspond precisely with standard definitions of IND-CPA and IND-CCA security for KEMs.

Game $\mathsf{MUC\text{-}IND}^b_{\mathsf{A},n}$	**Oracle** $\mathsf{Oenc}(j)$	**Oracle** $\mathsf{Odec}(j, c)$
00 for all $j \in [1\,..\,n]$:	05 $(K^0, c) \xleftarrow{\$} \mathsf{K.enc}(pk_j)$	09 if $c \in C_j$: return $\perp$
01 $\quad (pk_j, sk_j) \xleftarrow{\$} \mathsf{K.gen}()$	06 $K^1 \xleftarrow{\$} \mathcal{K}$	10 $K \leftarrow \mathsf{K.dec}(sk_j, c)$
02 $\quad C_j \leftarrow \emptyset$	07 $C_j \leftarrow C_j \cup \{c\}$	11 return K
03 $b' \xleftarrow{\$} \mathsf{A}(pk_1, \ldots, pk_n)$	08 return (K^b, c)	
04 return b'		

Figure 2: KEM security games $\mathsf{MUC\text{-}IND}^b_{\mathsf{A},n}$, $b \in \{0, 1\}$, modeling multi-user multi-challenge indistinguishability for n users. Adversary A can access oracles Oenc and Odec.

Akin to the PKE case, our KEM multi-user multi-challenge notion is equivalent with its single-user single-challenge relative—again up to a tightness loss linear in the number of users and the number of challenges. The corresponding proof is in Appendix B.2.

Lemma 3.2 *For any key-encapsulation mechanism* KEM*, any number of users* n*, and any adversary* A *that poses at most* q_e*-many* Oenc *and* q_d*-many* Odec *queries per user, there exists an adversary* B *such that* $\mathbf{Adv}^{\mathsf{muc\text{-}ind}}_{\mathsf{KEM},\mathsf{A},n} \leq n \cdot q_e \cdot \mathbf{Adv}^{\mathsf{muc\text{-}ind}}_{\mathsf{KEM},\mathsf{B},1}$*, where* B *poses at most one* Oenc *and* q_d*-many* Odec *queries. Further, the running time of* B *is at most that of* A *plus the time needed to perform* nq_e*-many* K.enc *operations and* nq_d*-many* K.dec *operations.*

DATA ENCAPSULATION. A data-encapsulation mechanism $\mathsf{DEM} = (\mathsf{D.enc}, \mathsf{D.dec})$ for a message space $\mathcal{M}$ is a pair of deterministic algorithms associated with a finite key space $\mathcal{K}$ and a ciphertext space $\mathcal{C}$. The encapsulation algorithm D.enc takes a key $K \in \mathcal{K}$ and a message $m \in \mathcal{M}$, and outputs a ciphertext $c \in \mathcal{C}$. The decapsulation algorithm D.dec takes a key $K \in \mathcal{K}$ and a ciphertext $c \in \mathcal{C}$, and outputs either a message $m \in \mathcal{M}$ or the special symbol $\perp \notin \mathcal{M}$ to indicate rejection. The correctness requirement is that for all $K \in \mathcal{K}$ and $m \in \mathcal{M}$ we have $\mathsf{D.dec}(K, \mathsf{D.enc}(K, m)) = m$.

As a security requirement for DEMs we formalize a multi-instance variant of the standard one-time indistinguishability notion: In our model the adversary can request one challenge encapsulation for each of a total of N independent keys; decapsulation queries are not restricted and can be asked multiple times for the same key. The corresponding games are in Figure 3. Note that lines 05 and 09 ensure that the adversary cannot ask for decapsulations with respect to a key before having a challenge message encapsulated with it. (This matches the typical situation as it emerges in a KEM/DEM hybrid.) For a scheme DEM, to any adversary A and any number of instances N we associate the distinguishing advantage $\mathbf{Adv}^{\mathsf{miot\text{-}ind}}_{\mathsf{DEM},\mathsf{A},N} := |\Pr[\mathsf{MIOT\text{-}IND}^0_{\mathsf{A},N}] - \Pr[\mathsf{MIOT\text{-}IND}^1_{\mathsf{A},N}]|$. Note that if Q_d specifies a global upper bound on the number of Odec queries, then the single-instance configurations $(N, Q_d) = (1, 0)$

and $(N, Q_d) = (1, \infty)$ correspond precisely with standard definitions of OT-IND-CPA and OT-IND-CCA security for DEMs.

Game $\mathsf{MIOT\text{-}IND}^b_{\mathsf{A},N}$	**Oracle** $\mathsf{Oenc}(j, m_0, m_1)$	**Oracle** $\mathsf{Odec}(j, c)$
00 for all $j \in [1 \,..\, N]$:	05 if $C_j \neq \emptyset$: return $\bot$	09 if $C_j = \emptyset$: return $\bot$
01 $\quad K_j \xleftarrow{\$} \mathcal{K}$	06 $c \leftarrow \mathsf{D.enc}(K_j, m_b)$	10 if $c \in C_j$: return $\bot$
02 $\quad C_j \leftarrow \emptyset$	07 $C_j \leftarrow C_j \cup \{c\}$	11 $m \leftarrow \mathsf{D.dec}(K_j, c)$
03 $b' \xleftarrow{\$} \mathsf{A}$	08 return c	12 return m
04 return b'		

Figure 3: DEM security games $\mathsf{MIOT\text{-}IND}^b_{\mathsf{A},N}$, $b \in \{0, 1\}$, modeling multi-instance one-time indistinguishability for N instances. Adversary A can access oracles Oenc and Odec.

Similarly to the cases of PKE and KEMs, our multi-instance notion for DEMs is equivalent to its single-instance counterpart, with a tightness loss of N. The corresponding proof is in Appendix B.3.

Lemma 3.3 *For any data-encapsulation mechanism* DEM*, any number of instances* N*, and any adversary* A *that poses at most* Q_d*-many* Odec *queries in total, there exists an adversary* B *such that* $\mathbf{Adv}^{\mathsf{miot\text{-}ind}}_{\mathsf{DEM},\mathsf{A},N} \leq N \cdot \mathbf{Adv}^{\mathsf{miot\text{-}ind}}_{\mathsf{DEM},\mathsf{B},1}$*, where* B *poses at most one* Oenc *and* Q_d*-many* Odec *queries. Further, the running time of* B *is at most that of* A *plus the time needed to perform* N*-many* $\mathsf{D.enc}$ *operations and* Q_d*-many* $\mathsf{D.dec}$ *operations.*

3.2 Hybrid encryption

The main application of KEMs and DEMs is the construction of public key encryption: To obtain a (hybrid) PKE scheme, a KEM is used to establish a session key and a DEM is used with this key to protect the confidentiality of the message [11]. The details of this construction are in Figure 4. It requires that the session key space of the KEM and the key space of the DEM coincide.

Proc $\mathsf{P.gen}$	**Proc** $\mathsf{P.enc}(pk, m)$	**Proc** $\mathsf{P.dec}(sk, \langle c_1, c_2 \rangle)$
00 $(pk, sk) \xleftarrow{\$} \mathsf{K.gen}$	02 $(K, c_1) \xleftarrow{\$} \mathsf{K.enc}(pk)$	05 $K \leftarrow \mathsf{K.dec}(sk, c_1)$
01 return (pk, sk)	03 $c_2 \leftarrow \mathsf{D.enc}(K, m)$	06 if $K = \bot$: return $\bot$
	04 return $\langle c_1, c_2 \rangle$	07 $m \leftarrow \mathsf{D.dec}(K, c_2)$
		08 return m

Figure 4: Hybrid construction of scheme PKE from schemes KEM and DEM. We write $\langle c_1, c_2 \rangle$ for the encoding of two ciphertext components into one.

The central composability result for hybrid encryption [11] says that if the KEM and DEM components are strong enough then also their combination is secure, with tight reduction. In Theorem 3.4 we give a generalized version of this claim: it considers multiple users and challenges, and implies the result from [11] as a corollary. Note that also our generalization allows for a tight reduction. The corresponding proof is in Appendix B.4.

Theorem 3.4 *Let* PKE *be the hybrid public-key encryption scheme constructed from a key-encapsulation mechanism* KEM *and a data-encapsulation mechanism* DEM *as in Figure 4. Then for any number of users* n *and any PKE adversary* A *that poses at most* q_e*-many* Oenc *and* q_d*-many* Odec *queries per user, there exist a KEM adversary* B *and a DEM adversary* C *such that*

$$\mathbf{Adv}^{\mathsf{muc\text{-}ind}}_{\mathsf{PKE},\mathsf{A},n} \leq 2\mathbf{Adv}^{\mathsf{muc\text{-}ind}}_{\mathsf{KEM},\mathsf{B},n} + \mathbf{Adv}^{\mathsf{miot\text{-}ind}}_{\mathsf{DEM},\mathsf{C},nq_e} \,.$$

The running time of B *is at most that of* A *plus the time required to run* nq_e *DEM encapsulations and* nq_e *DEM decapsulations. The running time of* C *is similar to the running time of* A *plus the time required to run* nq_e *KEM encapsulations,* nq_e *KEM decapsulations, and* nq_e *DEM decapsulations.* B *poses at most* q_e*-many* Oenc *and* q_d*-many* Odec *queries per user, and* C *poses at most* nq_e*-many* Oenc *and* nq_d*-many* Odec *queries in total.*

Theorem 3.4 bounds the distinguishing advantage of adversaries against hybrid PKE conditioned on its KEM and DEM components being secure. Note that from this result it cannot be deduced that deploying

an insecure DEM (potentially in combination with a secure KEM) necessarily leads to insecure PKE. We show in Theorem 3.5 that also the latter implication holds. To ease the analysis, instead of requiring MUC-IND-like properties of the KEM, we rather assume that it has uniformly distributed session keys. Formally this means that for all public keys pk the distribution of $[(K, c) \xleftarrow{\$} \mathsf{K.enc}(pk)$; output $K]$ is identical with the uniform distribution on key space $\mathcal{K}$. The proof is in Appendix B.5.

Theorem 3.5 *For a key-encapsulation mechanism* KEM *and a data-encapsulation mechanism* DEM *let* PKE *be the corresponding hybrid encryption scheme. If* KEM *has uniform keys in* $\mathcal{K}$*, any attack on* DEM *can be converted to an attack on* PKE*. More precisely, for any* n, q_e *and any DEM adversary* A *that poses in total at most* nq_e*-many* Odec *queries, there exists an adversary* B *such that*

$$\mathbf{Adv}^{\mathsf{miot\text{-}ind}}_{\mathsf{DEM},\mathsf{A},nq_e} \leq \mathbf{Adv}^{\mathsf{muc\text{-}ind}}_{\mathsf{PKE},\mathsf{B},n} + \frac{{nq_e}^2}{2\,|\mathcal{K}|} \ .$$

The running time of B *is about that of* A*, and* B *poses at most* q_e*-many* Oenc *queries per user and* Q_d*-many* Odec *queries in total.*

4 Deterministic DEMs and their multi-instance security

We give two generic key-collision attacks on the multi-instance security of (deterministic) DEMs. They have different attack goals (indistinguishability vs. key recovery) and succeed with slightly different probabilities. More precisely, in both cases the leading term of the success probability comes from the birthday bound and evaluates to roughly $N^2/\,|\mathcal{K}|$, and is thus much larger than the $N/\,|\mathcal{K}|$ that intuition might expect. By Theorem 3.5 the attacks can directly be lifted to ones targeting the multi-user multi-challenge security of a corresponding hybrid encryption scheme, achieving the same advantage.

4.1 A passive multi-instance distinguishing attack on DEMs

We describe an attack against multi-instance indistinguishability that applies generically to all DEMs. Notably, the attack is fully passive, i.e., the adversary does not pose any query to its Odec oracle. As technical requirements we assume a finite message space and a number of instances such that the inequalities $N^2 \leq 2\,|\mathcal{K}|$ and $|\mathcal{M}| \geq 3\,|\mathcal{K}| + N - 1$ are fulfilled. We consider these conditions extremely mild, since in practice $\mathcal{M}$ is very large and the value N can be chosen arbitrarily low by simply discarding some inputs.

For any value $N \in \mathbb{N}$ the details of our adversary $\mathsf{A} = \mathsf{A}_N$ are in Figure 5. It works as follows: It starts by picking uniformly at random messages $m_0, m_1^1, \ldots, m_1^N \in \mathcal{M}$ such that $m_1^1, \ldots, m_1^N$ are pairwise distinct. (Note the corresponding requirement $N \leq |\mathcal{M}|$ follows from above condition.) The adversary then asks for encapsulations of these messages in a way such that it obtains either N encapsulations of m_0 (if executed in game $\mathsf{MIOT\text{-}IND}^0$), or one encapsulation of each message m_1^j (if executed in game $\mathsf{MIOT\text{-}IND}^1$). If any two of the received ciphertexts collide, the adversary outputs 1; otherwise it outputs 0. The following theorem makes statements about advantage and running time of this adversary.

Adversary A_N
00 $m_0 \xleftarrow{\$} \mathcal{M}$
01 for all $j \in [1\,..\,N]$:
02 $\quad m_1^j \xleftarrow{\$} \mathcal{M} \setminus \{m_1^1, \ldots, m_1^{j-1}\}$
03 $\quad c^j \leftarrow \mathsf{Oenc}(j, m_0, m_1^j)$
04 return 1 iff $\mathbf{Coll}_2[c^1, \ldots, c^N]$

Figure 5: Adversary A against the multi-instance indistinguishability (MIOT-IND) of a DEM. It asks for N encapsulations (line 03) but does not use its decapsulation oracle.

Theorem 4.1 *For a finite message space* $\mathcal{M}$*, let* DEM *be a DEM with key space* $\mathcal{K}$*. Suppose that* $N^2 \leq 2\,|\mathcal{K}|$ *and* $|\mathcal{M}| \geq 3\,|\mathcal{K}| + N - 1$*. Then adversary* A *from Figure 5 breaks the* N*-instance indistinguishability of* DEM*, achieving an advantage of*

$$\mathbf{Adv}^{\mathsf{miot\text{-}ind}}_{\mathsf{DEM},\mathsf{A},N} \geq \frac{N(N-1)}{12\,|\mathcal{K}|} \ .$$

The adversary has a running time of $\mathcal{O}(N \log N)$, and poses N-many Oenc *and no* Odec *queries.*

We remark that, more generally, the bound on $|\mathcal{M}|$ can be relaxed to $|\mathcal{M}| \geq 2\,|\mathcal{K}|\,(1+\delta) + N - 1$ for some $\delta \geq 0$ to obtain $\mathbf{Adv}^{\mathsf{miot\text{-}ind}}_{\mathsf{DEM},\mathsf{A},N} \geq \frac{\delta}{\delta+1} \cdot \frac{N(N-1)}{4|\mathcal{K}|}$. (Theorem 4.1 is a special case with $\delta = 1/2$.)

Proof. The task of collecting N ciphertexts and checking for the occurrence of a collision can be completed in $\mathcal{O}(N \log N)$ operations. In the following we first assess the performance of the adversary when executed in games $\mathsf{MIOT\text{-}IND}^0$ and $\mathsf{MIOT\text{-}IND}^1$; then we combine the results.

Case $\mathsf{MIOT\text{-}IND}^0$. Adversary A receives N encapsulations of the same message m_0, created with N independently picked keys $K_1, \ldots, K_N$. If two of these keys collide the corresponding (deterministically computed) encapsulations collide as well and A returns 1. Since $N(N-1) < N^2 \leq 2\,|\mathcal{K}|$ by Lemma A.1 we obtain

$$\Pr[\mathsf{MIOT\text{-}IND}^0_{\mathsf{A},N}] \geq \frac{N(N-1)}{4\,|\mathcal{K}|}\ .$$

Case $\mathsf{MIOT\text{-}IND}^1$. Adversary A receives encapsulations $c^1, \ldots, c^N$ of uniformly distributed (but distinct) messages $m_1^1, \ldots, m_1^N$. Denote with K_j the key used to compute c^j, let $\mathcal{M}_j \coloneqq \mathcal{M} \setminus \{m_1^1, \ldots, m_1^{j-1}\}$, and let further $\mathcal{C}_j \coloneqq \mathsf{D.enc}(K_j, \mathcal{M}_j)$ denote the image of $\mathcal{M}_j$ under (injective) function $\mathsf{D.enc}(K_j, \cdot)$. Observe this setup implies $|\mathcal{C}_j| = |\mathcal{M}_j|$ and $|\mathcal{C}_1| > \ldots > |\mathcal{C}_N|$. If further follows that each ciphertext c^j is uniformly distributed in set $\mathcal{C}_j$.

We aim at establishing an upper-bound on the collision probability of ciphertexts $c^1, \ldots, c^N$. The maximum collision probability is attained in the worst-case $\mathcal{C}_1 \supset \ldots \supset \mathcal{C}_N$, in which it is bounded by the collision probability of choosing N values uniformly from a set of cardinality $|\mathcal{C}_N| = |\mathcal{M}| - N + 1$. Using again Lemma A.1 and $|\mathcal{M}| \geq 3\,|\mathcal{K}| + N - 1$ we obtain

$$\Pr[\mathsf{MIOT\text{-}IND}^1_{\mathsf{A},N}] \leq \frac{1}{2} \cdot \frac{N(N-1)}{|\mathcal{M}| - N + 1} \leq \frac{N(N-1)}{6\,|\mathcal{K}|}\ .$$

Combining the two bounds yields the claimed result:

$$\mathbf{Adv}^{\mathsf{miot\text{-}ind}}_{\mathsf{DEM},\mathsf{A},N} \geq \frac{N(N-1)}{12\,|\mathcal{K}|}\ .$$ □

4.2 A passive multi-instance key-recovery attack on DEMs

We give a generic attack on DEMs that aims at recovering keys rather than distinguishing encapsulations. Like in Section 4.1 the attack is passive. It is inspired by work of Zaverucha [24] and Chatterjee et al. [8]. However, our results are more general than theirs for being generic and not restricted to one specific DEM.

To formalize the notion of resilience against key recovery we correspondingly adapt the $\mathsf{MIOT\text{-}IND}$ game from Figure 3 and obtain the $\mathsf{MIOT\text{-}KR}$ game specified in Figure 6. The N-instance advantage of an adversary A is then defined as $\mathbf{Adv}^{\mathsf{miot\text{-}kr}}_{\mathsf{DEM},\mathsf{A},N} \coloneqq \Pr[\mathsf{MIOT\text{-}KR}_{\mathsf{A},N}]$. The following theorem establishes that for virtually all practical DEMs (including those based on CBC mode, CTR mode, OCB, etc., and even one-time pad encryption) there exist adversaries that achieve a considerable key recovery advantage, conditioned on the DEM key space being small enough. Concretely, the adversaries we propose encapsulate $2N$ times the same message (N times with random but known keys, and N times with random but unknown keys) and watch out for collisions of ciphertexts.[5] As any ciphertext collision stems (in practice) from a collision of keys, this method allows for key recovery.[6]

[5] While our setup is formally meaningful, in practice it would correspond to N parties, for a huge number N, encapsulating the same message m_0. This might feel rather unrealistic. However, we argue that a close variant of the attack might very well have the potential for practicality: All widely deployed DEMs are *online*, i.e., compute ciphertexts 'left-to-right'. For such DEMs, for our attack to be successful, it suffices that the N parties encapsulate (different) messages that have a common prefix, for instance a standard protocol header.

[6] The efficiency of this attack can likely be improved, on a heuristic basis, by deploying dedicated data structures like rainbow tables.

Game MIOT-KR$_{\mathsf{A},N}$	**Oracle** Oenc(j, m)	**Oracle** Odec(j, c)
00 for all $j \in [1 \,..\, N]$:	05 if $C_j \neq \emptyset$: return $\perp$	09 if $C_j = \emptyset$: return $\perp$
01 $\quad K_j \xleftarrow{\$} \mathcal{K}$	06 $c \leftarrow \mathsf{D.enc}(K_j, m)$	10 if $c \in C_j$: return $\perp$
02 $\quad C_j \leftarrow \emptyset$	07 $C_j \leftarrow C_j \cup \{c\}$	11 $m \leftarrow \mathsf{D.dec}(K_j, c)$
03 $(K, i) \xleftarrow{\$} \mathsf{A}$	08 return c	12 return m
04 return 1 iff $K = K_i$		

Figure 6: DEM security game MIOT-KR$_{\mathsf{A},N}$ modeling resilience against key recovery, for N instances. Adversary A is given access to oracles Oenc and Odec.

Theorem 4.2 *Fix a DEM and denote its key space with $\mathcal{K}$ and its message space with $\mathcal{M}$. Let $m_0 \in \mathcal{M}$ be any fixed message. Fixing $N \in \mathbb{N}$ as a parameter, consider the adversary $\mathsf{A} = \mathsf{A}_N$ specified in Figure 7. We then have*

$$\mathbf{Adv}^{\text{miot-kr}}_{\mathsf{DEM},\mathsf{A},N} \geq p(m_0) \cdot \min\left\{\frac{1}{2}, \frac{N^2}{2\,|\mathcal{K}|}\right\},$$

where $p(m_0)$ denotes the collision probability

$$p(m_0) \coloneqq \Pr_{K_1, K_2 \xleftarrow{\$} \mathcal{K}}[K_1 = K_2 \mid \mathsf{D.enc}(K_1, m_0) = \mathsf{D.enc}(K_2, m_0)]\,.$$

The adversary has a running time of $\mathcal{O}(N \log N)$, and poses N-many Oenc *and no* Odec *queries.*

We further prove that in the case of data encapsulation via one-time pad encryption we have $p(m_0) = 1$ for any m_0. Further, in the case of CBC-based encapsulation there exists a message m_0 such that $p(m_0) = |\mathcal{B}|\,/(|\mathcal{B}| + |\mathcal{K}| - 1)$, where $\mathcal{B}$ is the block space of the blockcipher and the latter is modeled as an ideal cipher.

Adversary A_N
00 for all $i \in [1 \,..\, N]$:
01 $\quad K_i \xleftarrow{\$} \mathcal{K} \setminus \{K_1, \ldots, K_{i-1}\}$
02 $\quad c_i \leftarrow \mathsf{D.enc}(K_i, m_0)$
03 for all $j \in [1 \,..\, N]$:
04 $\quad c'_j \leftarrow \mathsf{Oenc}(j, m_0)$
05 if $\exists (i, j) \in [1 \,..\, N]^2$ s.t. $c_i = c'_j$:
06 $\quad$ return (K_i, j)
07 return $\perp$

Figure 7: Adversary A against multi-instance key recovery (MIOT-KR) of a DEM. It asks for N challenge encapsulations (line 04) but does not use its decapsulation oracle.

Note that the performance of our attack crucially depends on the choice of message m_0, and that there does not seem to be a general technique for identifying good candidates. In particular, (artificial) DEMs can be constructed where $p(m_0)$ is small for some m_0 but large for others, or where $p(m_0)$ is small even for very long messages m_0.

Proof. The running time of A is upper bounded by the search for collisions in line 05, since all other operations require at most linear time in N. We estimate the time bound: The list $c_1, \ldots, c_N$ is sorted, requiring time $\mathcal{O}(N \log N)$. Searching an element in the ordered list requires $\mathcal{O}(\log N)$ time. Repeating for all N searches requires $\mathcal{O}(N \log N)$. Combining these observations yields our statement.

We claim that the probability that the adversary does not output $\perp$ (in symbols, $\mathsf{A}_N \not\Rightarrow \perp$) is lower bounded by:

$$\Pr[\mathsf{A}_N \not\Rightarrow \perp] \geq 1 - \left(1 - \frac{N}{|\mathcal{K}|}\right)^N. \tag{1}$$

Since the DEM is deterministic, the probability to find any collision in line 05 is larger than the probability that any of the distinct N keys generated in lines 00–02 collides with one of the N keys $\tilde{K}_1, \ldots, \tilde{K}_N$ used by the MIOT-KR game to encapsulate. We compute now the latter probability.

Let $K \in \{\tilde{K}_1, \ldots, \tilde{K}_N\}$. We know that the key K is generated uniformly in $\mathcal{K}$. Since the keys $K_1, \ldots, K_N$ are distinct and independently chosen we can write: $\Pr[K \in \{K_1, \ldots, K_N\}] = N/|\mathcal{K}|$. Moreover, since the keys $\tilde{K}_1, \ldots, \tilde{K}_N$ are generated independently of each other, Equation (1) follows.

Let now (i, j) be the indices for which the condition in line 05 is triggered, i.e., $c_i = c'_j$ and A_N outputs K_i. We can write:

$$\begin{aligned}\mathbf{Adv}^{\mathsf{miot\text{-}kr}}_{\mathsf{DEM},\mathsf{A},N} &= \Pr[\mathsf{A}_N \not\Rightarrow \perp] \cdot \Pr[K_i = \tilde{K}_j \mid \mathsf{A}_N \not\Rightarrow \perp] \\ &\geq \left(1 - \left(1 - \frac{N}{|\mathcal{K}|}\right)^N\right) \cdot p(m_0) \ .\end{aligned}$$

We apply to the previous inequality Lemma A.4 to obtain:

$$\mathbf{Adv}^{\mathsf{miot\text{-}kr}}_{\mathsf{DEM},\mathsf{A},N} \geq p(m_0) \cdot \left(1 - \left(1 - \frac{N}{|\mathcal{K}|}\right)^N\right) \geq p(m_0) \cdot \min\left\{\frac{1}{2}, \frac{N^2}{2\,|\mathcal{K}|}\right\} ,$$

which proves our main statement. □

We compute $p(m_0)$ for two specific DEMs (one-time pad and CBC mode) and choices of m_0. We formalize the argument for CBC by considering single-block messages. We note that one can apply the same argument to other modes of operation, e.g., CTR. For notational simplicity we omit the description of the probability space, that is, uniform choice of $K_1, K_2 \in \mathcal{K}$.

One-time pad. The one-time pad DEM encapsulation is given by combining a key $K \in \mathcal{K} = \{0,1\}^k$ with a message $m \in \mathcal{M} = \{0,1\}^k$ using the XOR operation. In this case, if two ciphertexts for the same message collide, the same key must have been used to encapsulate the message. Thus $p(m_0) = 1$ for all m_0.

CBC with an ideal cipher. CBC-based DEM encapsulation consists of encrypting the message using a blockcipher in CBC mode with the zero initialization vector (IV). In the following analysis we assume an idealized blockcipher (ideal cipher model) represented by E. Note that since the IV is zero, encapsulating a single-block message m_0 under the key K is equivalent to enciphering m_0 with E_K. Let $\mathcal{B}$ be the block space. First we observe that for any single-block message m_0 we have

$$\begin{aligned}&\Pr[\mathsf{E}_{K_1}(m_0) = \mathsf{E}_{K_2}(m_0)] \\ &= \Pr[K_1 = K_2] + \Pr[K_1 \neq K_2] \Pr[\mathsf{E}_{K_1}(m_0) = \mathsf{E}_{K_2}(m_0) \mid K_1 \neq K_2] \\ &= |\mathcal{K}|^{-1} + (1 - |\mathcal{K}|^{-1})\,|\mathcal{B}|^{-1} \ .\end{aligned}$$

We then use the previous equality to compute $p(m_0)$ from its definition:

$$\begin{aligned}p(m_0) &= \frac{\Pr[K_1 = K_2]}{\Pr[\mathsf{E}_{K_1}(m_0) = \mathsf{E}_{K_2}(m_0)]} \\ &= \frac{|\mathcal{K}|^{-1}}{|\mathcal{K}|^{-1} + (1 - |\mathcal{K}|^{-1})\,|\mathcal{B}|^{-1}} = \frac{|\mathcal{B}|}{|\mathcal{B}| + |\mathcal{K}| - 1} \ .\end{aligned}$$

As an example, if $|\mathcal{B}| \geq |\mathcal{K}|$ then $p(m_0) > 1/2$ for any single-block message m_0.

5 Augmented data encapsulation

In the previous sections we showed that all deterministic DEMs, including those that are widely used in practice, might be less secure than expected in the face of multi-instance attacks. We further showed that, in the setting of hybrid encryption, attacks on DEMs can be leveraged to attacks on the overall PKE. Given that the KEM+DEM paradigm is so important in practice, we next address the question of how this situation can be remedied. One option would of course be to increase the DEM key size (recall that good success probabilities in Theorems 4.1 and 4.2 are achieved only for not too large key spaces); however, increasing key sizes might not be a viable option in practical systems. (Potential reasons for this include that blockciphers like AES are slower with long keys than with short keys, and that ciphers

like 3DES do not support key lengths that have a comfortable 'multi-instance security margin' in the first place.) A second option would be to augment the input given to the DEM encapsulation routine by an additional value. This idea was already considered in [24, p. 16] where, with the intuition of increasing the 'entropy' available to the DEM, it was proposed to use a KEM ciphertext as an initialization vector (IV) of a symmetric encryption mode. However, [24] does not contain any formalization or security analysis of this idea, and so it cannot be taken as granted that this strategy actually works. (And indeed, we show in Section 6.3 that deriving the starting value of blockcipher-based counter mode encryption from a KEM ciphertext is not ameliorating the situation for attacks based on indistinguishability.)

We formally explore the additional-input proposal for the DEM in this section. More precisely, we study two approaches of defining an *augmented data encapsulation mechanism* (ADEM), where we call the additional input the *tag*. The syntax is the same in both cases, but the security properties differ: either (a) the DEM encapsulator receives as the tag an auxiliary random (but public) string, or (b) the encapsulator receives as additional input a nonce (a 'number used once'). In both cases the decapsulation oracle operates with respect to the tag also used for encapsulation. After formalizing this we prove the following results: First, if the tag space is large enough, ADEMs that expect a nonce can safely replace ADEMs that expect a uniform tag. Second, ADEMs that expect a uniform tag can be constructed from ADEMs that expect a nonce by applying a random oracle to the latter. Our third result is that the augmented variant of hybrid encryption remains (tightly) secure.

Augmented data encapsulation. An augmented data encapsulation mechanism $\mathsf{ADEM} = (\mathsf{A.enc}, \mathsf{A.dec})$ for a message space $\mathcal{M}$ is a pair of deterministic algorithms associated with a finite key space $\mathcal{K}$, a tag space $\mathcal{T}$, and a ciphertext space $\mathcal{C}$. The encapsulation algorithm $\mathsf{A.enc}$ takes a key $K \in \mathcal{K}$, a tag $t \in \mathcal{T}$, and a message $m \in \mathcal{M}$, and outputs a ciphertext $c \in \mathcal{C}$. The decapsulation algorithm $\mathsf{A.dec}$ takes a key $K \in \mathcal{K}$, a tag $t \in \mathcal{T}$, and a ciphertext $c \in \mathcal{C}$, and outputs either a message $m \in \mathcal{M}$ or the special symbol $\perp \notin \mathcal{M}$ to indicate rejection. The correctness requirement is that for all $K \in \mathcal{K}$ and $t \in \mathcal{T}$ and $m \in \mathcal{M}$ we have $\mathsf{A.dec}(K, t, \mathsf{A.enc}(K, t, m)) = m$.

Augmented data encapsulation with uniform tags. The first security notion we formalize assumes that each encapsulation operation uses a fresh and uniformly picked tag (note this imposes the technical requirement that the tag space be finite). More precisely, while the tag may become public after the encapsulation operation has completed, it may not be disclosed to the adversary before fixing the message to be encapsulated. We formalize this notion of uniform-tag multi-instance one-time indistinguishability for ADEMs via the games specified in Figure 8. For a scheme ADEM, to any adversary A and any number of instances N we associate the distinguishing advantage $\mathbf{Adv}^{\mathsf{u\text{-}miot\text{-}ind}}_{\mathsf{ADEM},\mathsf{A},N} := |\Pr[\mathsf{U\text{-}MIOT\text{-}IND}^0_{\mathsf{A},N}] - \Pr[\mathsf{U\text{-}MIOT\text{-}IND}^1_{\mathsf{A},N}]|$.

Game $\mathsf{U\text{-}MIOT\text{-}IND}^b_{\mathsf{A},N}$	**Oracle** $\mathsf{Oenc}(j, m_0, m_1)$	**Oracle** $\mathsf{Odec}(j, c)$
00 for all $j \in [1\,..\,N]$:	05 if $C_j \neq \emptyset$: return $\perp$	09 if $C_j = \emptyset$: return $\perp$
01 $\quad(K_j, t_j) \xleftarrow{\$} \mathcal{K} \times \mathcal{T}$	06 $c \leftarrow \mathsf{A.enc}(K_j, t_j, m_b)$	10 if $c \in C_j$: return $\perp$
02 $\quad C_j \leftarrow \emptyset$	07 $C_j \leftarrow C_j \cup \{c\}$	11 $m \leftarrow \mathsf{A.dec}(K_j, t_j, c)$
03 $b' \xleftarrow{\$} \mathsf{A}$	08 return (t_j, c)	12 return m
04 return b'		

Figure 8: ADEM security games $\mathsf{U\text{-}MIOT\text{-}IND}^b_{\mathsf{A},N}$, $b \in \{0, 1\}$, for N instances. Adversary A is given access to oracles Oenc and Odec. The tags in line 11 are the same as the ones in line 06.

Augmented data encapsulation with nonces. Our second security notion for ADEMs requires the tag provided to each encapsulation operation to be unique (across all instances). The tag can be generated using any possible method (e.g., using some global type of counter). We formalize the corresponding security notion of nonce-based multi-instance one-time indistinguishability for ADEMs via the games specified in Figure 9. For a scheme ADEM, to any adversary A and any number of instances N we associate the distinguishing advantage $\mathbf{Adv}^{\mathsf{n\text{-}miot\text{-}ind}}_{\mathsf{ADEM},\mathsf{A},N} := |\Pr[\mathsf{N\text{-}MIOT\text{-}IND}^0_{\mathsf{A},N}] - \Pr[\mathsf{N\text{-}MIOT\text{-}IND}^1_{\mathsf{A},N}]|$.

5.1 Relations between ADEMs with uniform and nonce tags

The two types of ADEMs we consider here can be constructed from each other. More concretely, the following lemma shows that if the tag space is large enough, ADEMs that expect a nonce can safely

Game $\mathsf{N\text{-}MIOT\text{-}IND}^b_{\mathsf{A},N}$	**Oracle** $\mathsf{Oenc}(j,t,m_0,m_1)$	**Oracle** $\mathsf{Odec}(j,c)$
00 $T \leftarrow \emptyset$	06 if $C_j \neq \emptyset$: return $\perp$	12 if $C_j = \emptyset$: return $\perp$
01 for all $j \in [1\,..\,N]$:	07 if $t \in T$: return $\perp$	13 if $c \in C_j$: return $\perp$
02 $\quad K_j \xleftarrow{\$} \mathcal{K}$	08 $T \leftarrow T \cup \{t\}$; $t_j \leftarrow t$	14 $m \leftarrow \mathsf{A.dec}(K_j, t_j, c)$
03 $\quad C_j \leftarrow \emptyset$	09 $c \leftarrow \mathsf{A.enc}(K_j, t_j, m_b)$	15 return m
04 $b' \xleftarrow{\$} \mathsf{A}$	10 $C_j \leftarrow C_j \cup \{c\}$	
05 return b'	11 return c	

Figure 9: ADEM security games $\mathsf{N\text{-}MIOT\text{-}IND}^b_{\mathsf{A},N}$, $b \in \{0,1\}$, for N instances. Adversary A is given access to oracles Oenc and Odec. The tags in line 14 are the same as the ones in line 09.

replace ADEMs that expect a uniform tag. The proof is in Appendix B.6.

Lemma 5.1 *Let* ADEM *be an augmented data encapsulation mechanism. If the cardinality of its tag space* $\mathcal{T}$ *is large enough and* ADEM *is secure with non-repeating tags, then it is also secure with random tags. More precisely, for any number of instances* N *and any adversary* A *there exist an adversary* B *that makes the same amount of queries such that* $\mathbf{Adv}^{\mathsf{u\text{-}miot\text{-}ind}}_{\mathsf{ADEM},\mathsf{A},N} \leq \mathbf{Adv}^{\mathsf{n\text{-}miot\text{-}ind}}_{\mathsf{ADEM},\mathsf{B},N} + N^2/(2\,|\mathcal{T}|)$*. The running time of the two adversaries is similar.*

The following simple lemma shows that ADEMs that expect a uniform tag can be constructed from ADEMs that expect a nonce by applying a random oracle to the latter. The proof is immediate since all queries to the random oracle have different input, thus the corresponding output is uniformly random and independently generated.

Lemma 5.2 *Let* $\mathsf{ADEM} = (\mathsf{A.enc}, \mathsf{A.dec})$ *be an augmented data encapsulation mechanism with tag space* $\mathcal{T}$*. Let* $H\colon \mathcal{T}' \to \mathcal{T}$ *denote a hash function, where* $\mathcal{T}'$ *is another tag space. Define* $\mathsf{ADEM}' = (\mathsf{A.enc}', \mathsf{A.dec}')$ *such that* $\mathsf{A.enc}'(K,t,m) := \mathsf{A.enc}(K, H(t), m)$ *and* $\mathsf{A.dec}'(K,t,c) := \mathsf{A.dec}(K, H(t), c)$*. Then if* H *is modeled as a random oracle and if* ADEM *is secure with random tags in* $\mathcal{T}$*, then* ADEM' *is secure with non-repeating tags in* $\mathcal{T}'$*. More precisely, for any number of instances* N *and any adversary* A *there exists an adversary* B *with* $\mathbf{Adv}^{\mathsf{u\text{-}miot\text{-}ind}}_{\mathsf{ADEM},\mathsf{A},N} = \mathbf{Adv}^{\mathsf{n\text{-}miot\text{-}ind}}_{\mathsf{ADEM}',\mathsf{B},N}$*.*

5.2 Augmented hybrid encryption

A KEM and an ADEM can be combined to obtain a PKE scheme: the KEM establishes a session key and a first ciphertext component, and the ADEM is used on input the session key and the first ciphertext component (as tag) to protect the confidentiality of the message, creating a second ciphertext component. Figure 10 details this *augmented hybrid encryption*. It requires that the session key space of the KEM and the key space of the ADEM coincide. Further, the ciphertext space of the KEM needs to be a subset of the tag space of the ADEM.

Proc $\mathsf{P.gen}$	**Proc** $\mathsf{P.enc}(pk, m)$	**Proc** $\mathsf{P.dec}(sk, \langle c_1, c_2\rangle)$
00 $(pk, sk) \xleftarrow{\$} \mathsf{K.gen}$	02 $(K, c_1) \xleftarrow{\$} \mathsf{K.enc}(pk)$	05 $K \leftarrow \mathsf{K.dec}(sk, c_1)$
01 return (pk, sk)	03 $c_2 \leftarrow \mathsf{A.enc}(K, c_1, m)$	06 if $K = \perp$: return $\perp$
	04 return $\langle c_1, c_2\rangle$	07 $m \leftarrow \mathsf{A.dec}(K, c_1, c_2)$
		08 return m

Figure 10: Augmented hybrid construction of scheme PKE from schemes KEM and ADEM. We write $\langle c_1, c_2\rangle$ for the encoding of two ciphertext components into one.

The claim is that augmented hybrid encryption is more robust against attacks involving multiple users and challenges than standard hybrid encryption (see Figure 4) is. The security condition posed on the ADEM would be that it be secure when operated with nonces, and the security property posed on the KEM would be that it be both indistinguishable and have non-repeating ciphertexts (i.e., invoking its encapsulation algorithm twice on the same or different public keys does virtually never result in colliding ciphertexts). Technically, the latter property is implied by indistinguishability. However, to obtain better bounds, we formalize it as a statistical condition: To any scheme KEM we assign the maximum ciphertext-collision probability

$$p := \max_{pk_1, pk_2} \Pr[(K_1, c_1) \xleftarrow{\$} \mathsf{K.enc}(pk_1); (K_2, c_2) \xleftarrow{\$} \mathsf{K.enc}(pk_2) : c_1 = c_2]\,,$$

where the maximum is over all pairs pk_1, pk_2 of (potentially coinciding) public keys. Note that practical KEMs (ElGamal, RSA-based, Cramer–Shoup, ...) have much larger ciphertexts than session keys[7], so that the ciphertext-collision probability will always be negligible in practice. We proceed with a security claim for augmented hybrid encryption. The proof is in Appendix B.7.

Lemma 5.3 *Let* PKE *be the hybrid public-key encryption scheme constructed from a key-encapsulation mechanism* KEM *and an augmented data-encapsulation mechanism* ADEM *as in Figure 10. Let p be the maximum ciphertext-collision probability of* KEM *over all possible public keys. Then for any number of users n and any PKE adversary* A *that poses at most q_e-many* Oenc *and q_d-many* Odec *queries per user, there exist a KEM adversary* B *and an ADEM adversary* C *such that*

$$\mathbf{Adv}^{\mathsf{muc\text{-}ind}}_{\mathsf{PKE},\mathsf{A},n} \leq 2\mathbf{Adv}^{\mathsf{muc\text{-}ind}}_{\mathsf{KEM},\mathsf{B},n} + \mathbf{Adv}^{\mathsf{n\text{-}miot\text{-}ind}}_{\mathsf{ADEM},\mathsf{C},N} + 2\binom{N}{2}p\ ,$$

where $N = nq_e$. The running time of B *is at most that of* A *plus the time required to run nq_e ADEM encapsulations and nq_e ADEM decapsulations. The running time of* C *is similar to the running time of* A *plus the time required to run nq_e KEM encapsulations, nq_e KEM decapsulations, and nq_e ADEM decapsulations.* B *poses at most q_e-many* Oenc *and q_d-many* Odec *queries per user, and* C *poses at most nq_e-many* Oenc *and nq_d-many* Odec *queries in total.*

6 Constructions of augmented data encapsulation

We construct two augmented data-encapsulation mechanisms and analyze their security. The schemes are based on operating a function in counter mode. If the function is instantiated with an ideal random function then the ADEMs are secure beyond the birthday bound. (We also show that if the function is instead instantiated with an idealized blockcipher, i.e., a random permutation, the schemes' security may degrade.) Practical candidates for instantiating the ideal random function are for instance the compression functions of standardized Merkle–Damgård hash functions, e.g., of SHA2.[8][9] Another possibility is deriving the random function from an ideal cipher as in [21].

6.1 Counter-mode encryption

Many practical DEMs are based on operating a blockcipher E in counter mode (CTR). Here, in brief, the encapsulation key is used as the blockcipher key, a sequence of message-independent input blocks is enciphered under that key, and the output blocks are XOR-ed into the message. More concretely, if under some key K a message m shall be encapsulated that, without requiring padding, evenly splits into blocks $v_1 \| \ldots \| v_l$, then the DEM ciphertext is the concatenation $w_1 \| \ldots \| w_l$ where $w_i = v_i \oplus E_K(i)$.

In the context of this paper, three properties of this way of constructing a DEM are worth pointing out: (a) the 'counting' component of CTR mode serves effectively only one purpose: preventing that any two inputs to the blockcipher are the same; (b) any 'starting value' for the counter can be used; (c) security analyses of CTR mode typically model E as a pseudorandom function (as opposed to a pseudorandom permutation)[10].

In Figure 11 we detail three ways of turning the principles of CTR mode into a DEM encapsulation routine. In all cases the underlying primitive is, syntactically, a function $F\colon \mathcal{K} \times \mathcal{B} \to \mathcal{D}$ that takes a key $K \in \mathcal{K}$ and maps some finite input space $\mathcal{B}$ into some finite group $(\mathcal{D}, \oplus)$. (Intuitively, $\mathcal{B}$ serves as a space of input blocks derived from a counter, and $\mathcal{D}$ as a space of pads that can be XORed into message blocks; note that if F is instantiated with a blockcipher we have $\mathcal{B} = \mathcal{D}$, but we explicitly allow also other instantiations.) The most basic encapsulation routine based on CTR mode that we consider, and the one closest to what we sketched above, is CTR0enc. Note that this DEM further assumes a bijection $[\![\cdot]\!]_L\colon \mathbb{Z}/L\mathbb{Z} \to \mathcal{L}$ with $\mathcal{L} = \mathcal{B}$. (Intuitively, this bijection turns a counter that is cyclic with period length L into input blocks for F; see Section 2 for the notation.) We finally point out that all three variants of CTR mode that we formalize exclusively work with fixed-length multi-block messages (i.e.,

[7] This is no coincidence but caused by generic attacks against cyclic groups, RSA, etc.

[8] These compression functions are regularly modeled as having random behavior [2, 13].

[9] The idea to construct a DEM from a hash function's compression function already appeared in the OMD schemes from [9].

[10] Technically, the PRP/PRF switching lemma [5] measures the price one has to pay for pursuing this modeling approach.

$\mathcal{M} = \mathcal{D}^l$). This choice, that we made for simplicity of exposition, is not really a restriction as 'any-length' CTR mode encryption can be simulated from 'block-wise' CTR mode encryption.

Proc CTR0enc(K, m)	**Proc** CTR+enc(K, t, m)	**Proc** CTR‖enc(K, t, m)
00 $(v_1, \ldots, v_l) \leftarrow m$	05 $(v_1, \ldots, v_l) \leftarrow m$	10 $(v_1, \ldots, v_l) \leftarrow m$
01 for all $i \in [1 \,..\, l]$:	06 for all $i \in [1 \,..\, l]$:	11 for all $i \in [1 \,..\, l]$:
02 $\quad w_i \leftarrow v_i \oplus F(K, [\![i]\!]_L)$	07 $\quad w_i \leftarrow v_i \oplus F(K, [\![t+i]\!]_L)$	12 $\quad w_i \leftarrow v_i \oplus F(K, t \| [\![i]\!]_L)$
03 $c \leftarrow (w_1, \ldots, w_l)$	08 $c \leftarrow (w_1, \ldots, w_l)$	13 $c \leftarrow (w_1, \ldots, w_l)$
04 return c	09 return c	14 return c

Figure 11: Encapsulation algorithms of the CTR0 DEM, the CTR+ ADEM, and the CTR‖ ADEM, for multi-block messages. In CTR0enc and CTR+enc we assume $[\![\cdot]\!]_L \colon \mathbb{Z}/L\mathbb{Z} \to \mathcal{L}$ with $\mathcal{L} = \mathcal{B}$, and in CTR‖enc we assume $[\![\cdot]\!]_L \colon \mathbb{Z}/L\mathbb{Z} \to \mathcal{L}$ and $\mathcal{T}$ such that $\mathcal{B} = \mathcal{T} \times \mathcal{L}$. In all cases, finding the corresponding decapsulation routines is immediate.

The two remaining procedures in Figure 11 are ADEM encapsulation routines. The first one, CTR+enc, is the natural variant of CTR0enc where the tag space is $\mathcal{T} = [1 \,..\, L]$ and the tag specifies the starting value of the counter. The second, CTR‖enc, concatenates tag and counter. Here, the tag space $\mathcal{T}$ and parameter space $\mathcal{L}$ have to be arranged such that $\mathcal{B} = \mathcal{T} \times \mathcal{L}$.

We analyze the security of CTR+ and CTR‖ in the upcoming sections. Scheme CTR0 is not an ADEM and falls prey to our earlier attacks.

6.2 Security of function-based counter mode

We establish upper bounds on the advantage of U-MIOT-IND adversaries against the CTR+ and CTR‖ ADEMs.

6.2.1 Counter mode with tag-controlled starting value

We limit the maximum amount of blocks in an encapsulation query to a fixed value ℓ. Prerequisites to our statement on CTR+ are two conditions on the number of instances relative to $\mathcal{K}$ and $\mathcal{T} = [1 \,..\, L]$. The bound is namely $N \leq \min\left\{ |\mathcal{K}|^{1/2}, (|\mathcal{T}|/(2\ell))^{1/(1+\delta)} \right\}$, for some arbitrary constant δ such that $1/N \leq \delta \leq 1$. Despite this restriction we consider our statement to be reflecting real-world applications: As an extreme example we see that the values $|\mathcal{K}| = |\mathcal{T}| = 2^{128}$, $N = 2^{56}$, $\ell = 2^{56}$, $q = 2^{64}$ and $\delta = 2/7$ fit above condition, yielding a maximum advantage of around 2^{-61}.

Theorem 6.1 *Suppose $N \leq \min\left\{ |\mathcal{K}|^{1/2}, (|\mathcal{T}|/(2\ell))^{1/(1+\delta)} \right\}$, for some $1/N \leq \delta \leq 1$, and suppose that F is modeled as a random oracle (using oracle F). Then for any adversary A against N-instance uniform-tag indistinguishability of CTR+ that poses at most q queries to F, no decapsulation queries, and encapsulates messages of length at most ℓ blocks we have:*

$$\mathbf{Adv}^{\text{u-miot-ind}}_{\text{CTR+},\mathsf{A},N} \leq \frac{1}{3}\frac{N}{|\mathcal{K}|} + \frac{4\ell - 2}{|\mathcal{T}|} + \frac{2q}{|\mathcal{K}|}\left(1 + \frac{1}{\delta}\right).$$

The core of the proof exploits that the outputs of (random oracle) F that are used to encapsulate are uniformly distributed in $\mathcal{D}$ and independent of each other. This requires forcing the inputs to be distinct in $\mathcal{L}$. We give further insight on some non-standard techniques the we use in the analysis in the proof.

Proof of Theorem 6.1. The definition of the games $\mathsf{G}^{0,b}_{\mathsf{A},N}$, $\mathsf{G}^{1,b}_{\mathsf{A},N}$, $\mathsf{G}^{2,b}_{\mathsf{A},N}$ and $\mathsf{G}^{3,b}_{\mathsf{A},N}$ are found in Figure 12. Except for some bookkeeping, game $\mathsf{G}^{0,b}_{\mathsf{A},N}$ is equivalent to game $\mathsf{U\text{-}MIOT\text{-}IND}^b_{\mathsf{A},N}$, where $b \in \{0, 1\}$. For $j \in [1 \,..\, N]$ we define $T_j = [\![t_j \twoheadrightarrow \ell]\!]_L$.

Game G^1. In game $\mathsf{G}^{1,b}_{\mathsf{A},N}$ we implicitly generate pairs of colliding keys. We loop over all pairs (j_1, j_2) such that $1 \leq j_1 < j_2 \leq N$. If both indices were not previously paired ($\mathsf{matched}[j_1] = \mathsf{matched}[j_2] = \mathit{false}$) and the corresponding keys collide ($K_{j_1} = K_{j_2}$) then the two indices are marked as paired. Moreover, if the corresponding tag ranges collide ($T_{j_1} \cap T_{j_2} \neq \emptyset$) the flag bad_1 in line 10 is raised and the game aborts. We claim that

$$|\Pr[\mathsf{G}^{0,b}_{\mathsf{A},N}] - \Pr[\mathsf{G}^{1,b}_{\mathsf{A},N}]| \leq \Pr[\mathrm{bad}_1] \leq \frac{2\ell - 1}{|\mathcal{T}|}. \tag{2}$$

To prove (2), we want to compute the probability $\Pr[\text{bad}_1]$. Let m_{pairs} be the number of colliding key pairs in game $\mathsf{G}^{1,b}_{\mathsf{A},N}$, i.e., $2m_{\text{pairs}}$ entries of flag matched are set to 1 at the end of the game. Then, for every $0 \le i \le \lfloor N/2 \rfloor$,

$$\Pr[\text{bad}_1 \mid m_{\text{pairs}} = i] \le (2\ell - 1)i / |\mathcal{T}| \ .$$

This follows from the independent choices of the values K_j, t_j for each instance $j \in [1\,..\,N]$, and because for each pair of indices $j_1, j_2 \in [1\,..\,N], j_1 \ne j_2$, and for any choice of t_{j_1} there are exactly $2\ell - 1$ possible values of t_{j_2} such that $T_{j_1} \cap T_{j_2} \ne \emptyset$.

The sets $\{m_{\text{pairs}} = i\}$, $i \in 0, \ldots, \lfloor N/2 \rfloor$, partition the probability space, thus:

$$\begin{aligned}\Pr[\text{bad}_1] &= \sum_{i=0}^{\lfloor N/2 \rfloor} \Pr[\text{bad}_1 \mid m_{\text{pairs}} = i] \Pr[m_{\text{pairs}} = i] \\ &\le \frac{2\ell - 1}{|\mathcal{T}|} \sum_{i=0}^{\lfloor N/2 \rfloor} i \Pr[m_{\text{pairs}} = i] = \frac{2\ell - 1}{|\mathcal{T}|} \sum_{i=1}^{\lfloor N/2 \rfloor} \Pr[m_{\text{pairs}} \ge i] \ . \end{aligned} \tag{3}$$

The last equality follows since the expected value of any random variable m with values in $\mathbb{N}$ can be written as $\sum_{i=0}^{\infty} i \Pr[m = i] = \sum_{i=1}^{\infty} \Pr[m \ge i]$. We show by induction that the terms of the sum are:

$$p_i \coloneqq \Pr[m_{\text{pairs}} \ge i] \le \left(\frac{N^2}{2\,|\mathcal{K}|} \right)^i \ . \tag{4}$$

To prove (4), we consider a slightly different event. We say that *key K_i is bad* if $K_j = K_i$ for some $1 \le i < j$. Let m_{badkeys} be the random variable counting the number of bad keys.

Since every colliding key pair implies at least one bad key, then by Lemma A.5:

$$\Pr[m_{\text{pairs}} \ge i] \le \Pr[m_{\text{badkeys}} \ge i] \le \left(\frac{N^2}{2\,|\mathcal{K}|} \right)^i \ .$$

Finally we prove (2) by combining (3) and (4), and by observing that from our hypothesis $N^2 / |\mathcal{K}| \le 1$:

$$\Pr[\text{bad}_1] \le \frac{2\ell - 1}{|\mathcal{T}|} \sum_{i=1}^{\lfloor N/2 \rfloor} \left(\frac{N^2}{2\,|\mathcal{K}|} \right)^i \le \frac{2\ell - 1}{|\mathcal{T}|} \sum_{i=1}^{\infty} \frac{1}{2^i} = \frac{2\ell - 1}{|\mathcal{T}|} \ . \tag{5}$$

Game G^2. Game $\mathsf{G}^{2,b}_{\mathsf{A},N}$ is equivalent to $\mathsf{G}^{1,b}_{\mathsf{A},N}$, with the exception that it raises flag bad_2 in line 12 and aborts if any three keys collide. By Lemma A.2, and since $N^2 / |\mathcal{K}| \le 1$, we obtain

$$|\Pr[\mathsf{G}^{1,b}_{\mathsf{A},N}] - \Pr[\mathsf{G}^{2,b}_{\mathsf{A},N}]| \le \Pr[\text{bad}_2] \le \frac{1}{6} \frac{N^3}{|\mathcal{K}|^2} \le \frac{1}{6} \frac{N}{|\mathcal{K}|} \ . \tag{6}$$

Game G^3. Game $\mathsf{G}^{3,b}_{\mathsf{A},N}$ is equivalent to $\mathsf{G}^{2,b}_{\mathsf{A},N}$, with the exception that the game raises flag bad_3 in line 23 and aborts if A makes a query (K, v) to F for which there exists an index $j \in [1\,..\,N]$ such that $K = K_j$ and $v \in T_j$. In the following we fix m_{inters} to be the random variable that counts the maximum number of sets $T_1, \ldots, T_N$ whose intersection is non-empty.

Fix a query (K, v) to F. For each $i \in [1\,..\,N]$ we have $\Pr[\exists j \in [1\,..\,N] : v \in T_j \wedge K = K_j \mid m_{\text{inters}} = i] \le i / |\mathcal{K}|$, because in the worst case v belongs to exactly m_{inters} of the sets $T_1, \ldots, T_N$. This bound yields

$$\begin{aligned}&\Pr[\exists j \in [1\,..\,N] : v \in T_j \wedge K = K_j] \\ &= \sum_{i=1}^{N} \Pr[\exists j \in [1\,..\,N] : v \in T_j \wedge K = K_j \mid m_{\text{inters}} = i] \cdot \Pr[m_{\text{inters}} = i] \\ &\le \sum_{i=1}^{N} \frac{i}{|\mathcal{K}|} \cdot \Pr[m_{\text{inters}} = i] = \frac{1}{|\mathcal{K}|} \cdot \sum_{i=1}^{N} \Pr[m_{\text{inters}} \ge i] \ . \end{aligned} \tag{7}$$

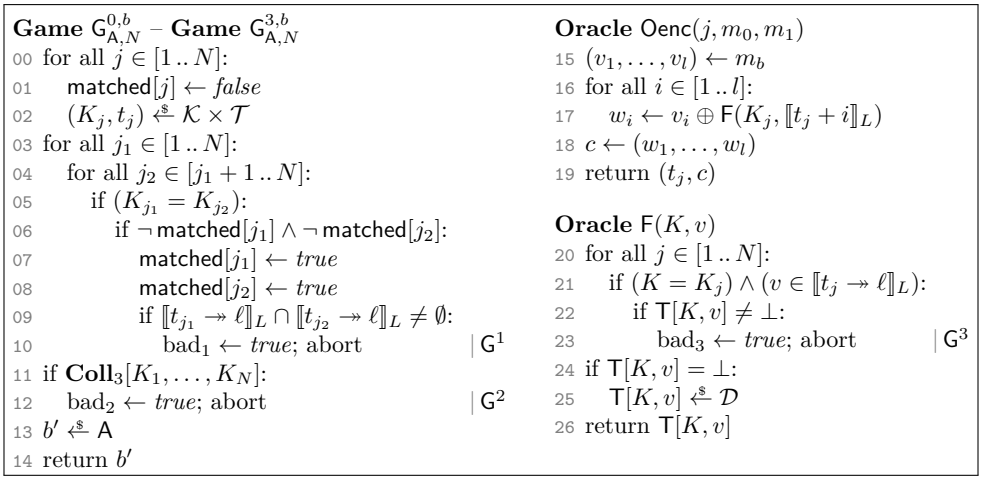

Figure 12: The security game $\mathsf{G}^{0,b}_{\mathsf{A},N}$ for CTR+ in the random oracle model, and the games $\mathsf{G}^{1,b}_{\mathsf{A},N}$, $\mathsf{G}^{2,b}_{\mathsf{A},N}$, and $\mathsf{G}^{3,b}_{\mathsf{A},N}$. Adversary A is given access to oracles Oenc and F, and can query the oracle Oenc at most once for the same index j.

By Lemma A.3 we have $\Pr[m_{\text{inters}} \geq i+1] \leq N^{i+1}\ell^i / |\mathcal{T}|^i$. For all $i \geq 1/\delta$ these values can be upper bounded by $\frac{N^{i+1}\ell^i}{|\mathcal{T}|^i} \leq \left(\frac{N^{1+\delta}\ell}{|\mathcal{T}|}\right)^i \leq \frac{1}{2^i}$. Thus we can split the sum (7) into

$$\begin{aligned}\frac{1}{|\mathcal{K}|} \cdot \sum_{i=1}^{N} \Pr[m_{\text{inters}} \geq i] &\leq \frac{1}{|\mathcal{K}|}\left(\sum_{i=1}^{\lfloor 1/\delta \rfloor} \Pr[m_{\text{inters}} \geq i] + \sum_{i=\lfloor 1/\delta \rfloor + 1}^{\infty} \frac{1}{2^{i-1}}\right) \\ &\leq \frac{1}{|\mathcal{K}|}\left(\frac{1}{\delta} + 1\right).\end{aligned}$$

Since m_{inters} is constant for all q queries to F, a union bound gives us

$$|\Pr[\mathsf{G}^{2,b}_{\mathsf{A},N}] - \Pr[\mathsf{G}^{3,b}_{\mathsf{A},N}]| \leq \Pr[\mathrm{bad}_3] \leq \frac{q}{|\mathcal{K}|}\left(\frac{1}{\delta} + 1\right). \tag{8}$$

The theorem follows by combining the bounds in (2), (6), (8) for both $b = 0$ and $b = 1$ and the fact that game $\mathsf{G}^{3,b}_{\mathsf{A},N}$ is independent of the bit b. □

6.2.2 Counter mode with tag prefix

We have the following security statement on CTR‖. Note it is slightly better than the one for CTR+.

Theorem 6.2 *Suppose $N \leq \min\left\{|\mathcal{K}|^{1/2}, (|\mathcal{T}|/2)^{1/(1+\delta)}\right\}$, for some $1/N \leq \delta \leq 1$, and suppose that F is modeled as a random oracle (using oracle F). Then for any adversary A against N-instance uniform-tag indistinguishability of CTR‖ that poses at most q queries to F and no decapsulation queries we have:*

$$\mathbf{Adv}^{\text{u-miot-ind}}_{\mathsf{CTR}\|,\mathsf{A},N} \leq \frac{1}{3}\frac{N}{|\mathcal{K}|} + \frac{1}{|\mathcal{T}|} + \frac{2q}{|\mathcal{K}|}\left(1 + \frac{1}{\delta}\right).$$

Proof. We refer to Figure 13 for the definition of the games $\mathsf{G}^{0,b}_{\mathsf{A},N}$, $\mathsf{G}^{1,b}_{\mathsf{A},N}$, $\mathsf{G}^{2,b}_{\mathsf{A},N}$ and $\mathsf{G}^{3,b}_{\mathsf{A},N}$. Except for some bookkeeping, game $\mathsf{G}^{0,b}_{\mathsf{A},N}$ is equivalent to the security game $\mathsf{U\text{-}MIOT\text{-}IND}^b_{\mathsf{A},N}$, with $b \in \{0,1\}$.

Game G^1. Game $\mathsf{G}^{1,b}_{\mathsf{A},N}$ is equivalent to $\mathsf{G}^{0,b}_{\mathsf{A},N}$, except when any three keys collide. By Lemma A.2, and since $N^2/|\mathcal{K}| \leq 1$, we obtain

$$|\Pr[\mathsf{G}^{0,b}_{\mathsf{A},N}] - \Pr[\mathsf{G}^{1,b}_{\mathsf{A},N}]| \leq \Pr[\mathrm{bad}_1] \leq \frac{1}{6}\frac{N^3}{|\mathcal{K}|^2} \leq \frac{1}{6}\frac{N}{|\mathcal{K}|}. \tag{9}$$

Game G^2. In game $\mathsf{G}^{2,b}_{\mathsf{A},N}$ we abort when two events occur simultaneously: a key 2-collision and a collision of the corresponding tags. The probability to abort is by Lemma A.2, the independence of the two events, and the condition $N^2/|\mathcal{K}| \leq 1$:

$$|\Pr[\mathsf{G}^{1,b}_{\mathsf{A},N}] - \Pr[\mathsf{G}^{2,b}_{\mathsf{A},N}]| \leq \Pr[\text{bad}_2] \leq \frac{N^2}{2\,|\mathcal{K}|}\frac{1}{|\mathcal{T}|} \leq \frac{1}{2}\frac{1}{|\mathcal{T}|}\,. \tag{10}$$

Game G^3. Game $\mathsf{G}^{3,b}_{\mathsf{A},N}$ is equivalent to $\mathsf{G}^{2,b}_{\mathsf{A},N}$, with the exception that the game raises flag bad_3 in line 16 if some specific condition is met. To get an upper bound on the probability to distinguish $\mathsf{G}^{2,b}_{\mathsf{A},N}$ and $\mathsf{G}^{3,b}_{\mathsf{A},N}$ we compute the probability that the adversary explicitly queries F for an input $(K, v\|[\![i]\!]_L)$ such that for some $j \in [1\,..\,N]$, $K = K_j$ and $v = t_j$. This leads to the equation:

$$|\Pr[\mathsf{G}^{2,b}_{\mathsf{A},N}] - \Pr[\mathsf{G}^{3,b}_{\mathsf{A},N}]| \leq \Pr[\text{bad}_3] \leq \frac{q}{|\mathcal{K}|}\left(\frac{1}{\delta}+1\right). \tag{11}$$

Fix a query $(K, v\|[\![i]\!]_L)$ to F. Since the adversary knows all possible values of v used by Oenc after each call, the adversary must only guess the key. Assume that there are at most m_{coll} keys that use the same tag value v. Then the probability that flag bad_3 is triggered during this query is in the worst case $m_{\text{coll}}/|\mathcal{T}|$. We compute the probability of this event as follows.

$$\begin{aligned}
&\Pr[\exists j \in [1\,..\,N] : v = t_j \wedge K = K_j] \\
&= \sum_{i=1}^{N} \Pr[\exists j \in [1\,..\,N] : v = t_j \wedge K = K_j \mid m_{\text{coll}} = i] \cdot \Pr[m_{\text{coll}} = i] \\
&\leq \sum_{i=1}^{N} \frac{i}{|\mathcal{K}|} \cdot \Pr[m_{\text{coll}} = i] = \frac{1}{|\mathcal{K}|} \cdot \sum_{i=1}^{N} \Pr[m_{\text{coll}} \geq i]\,.
\end{aligned} \tag{12}$$

The last equality follows since the expected value of any random variable m with values in $\mathbb{N}$ can be written as $\sum_{i=0}^{\infty} i \Pr[m = i] = \sum_{i=1}^{\infty} \Pr[m \geq i]$.

Now we estimate the probability $\Pr[m_{\text{coll}} \leq i]$. Assume that $i \geq 1/\delta$. Then from Lemma A.2 and the condition $N \leq (|\mathcal{T}|/2)^{1/(1+\delta)}$ we can write:

$$\Pr[m_{\text{coll}} \geq i+1] \leq \frac{N^{i+1}}{(i+1)!\,|\mathcal{T}|^i} \leq \left(\frac{N^{1+\delta}}{|\mathcal{T}|}\right)^i \leq \frac{1}{2^i}\,.$$

Considering this observation we split the sum in Equation (12) into

$$\begin{aligned}
\frac{1}{|\mathcal{K}|} \cdot \sum_{i=1}^{N} \Pr[m_{\text{coll}} \geq i] &\leq \frac{1}{|\mathcal{K}|}\left(\sum_{i=1}^{\lfloor 1/\delta \rfloor} \Pr[m_{\text{coll}} \geq i] + \sum_{i=\lfloor 1/\delta \rfloor + 1}^{\infty} \frac{1}{2^{i-1}}\right) \\
&\leq \frac{1}{|\mathcal{K}|}\left(\frac{1}{\delta}+1\right).
\end{aligned}$$

Since m_{coll} is constant for all queries to F, a union bound yields our claim:

$$\Pr[\text{bad}_3] \leq \frac{q}{|\mathcal{K}|}\left(\frac{1}{\delta}+1\right).$$

The theorem follows by combining the bounds in (9), (10), (11) for both $b = 0$ and $b = 1$ and the fact that game $\mathsf{G}^{3,b}_{\mathsf{A},N}$ is independent of b. □

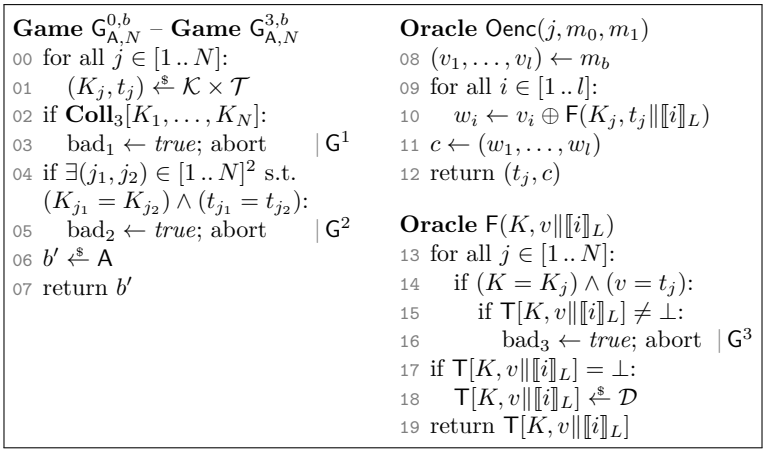

Figure 13: The security game $\mathsf{G}^{0,b}_{\mathsf{A},N}$ for $\mathsf{CTR}\|$ in the random oracle model, and the games $\mathsf{G}^{1,b}_{\mathsf{A},N}$, $\mathsf{G}^{2,b}_{\mathsf{A},N}$, and $\mathsf{G}^{3,b}_{\mathsf{A},N}$. Adversary A is given access to oracles Oenc and F, and can query the oracle Oenc at most once for the same index j.

6.3 On the security of permutation-based counter mode

In above Theorem 6.1 we assessed the security of the CTR+ ADEM, defined in respect to a function $F\colon \mathcal{K} \times \mathcal{B} \to \mathcal{D}$. The analysis modeled F as an ideal random function and showed that using sets $\mathcal{K}$ and $\mathcal{B}$ of moderate size (e.g., of cardinality 2^{128}) is sufficient to let CTR+ achieve security. We next show that if F is instead instantiated with a blockcipher and modeled as an ideal family of permutations, then the minimum cardinality of $\mathcal{B} = \mathcal{D}$ for achieving security is considerably increased (e.g., to values around 2^{256}).

Our argument involves the analysis of a U-MIOT-IND adversary A that is specified in Figure 14. Effectively, the idea of the attack is exploiting the tightness gap of the PRP/PRF switching lemma [5] via the multi-instance setting. More concretely, the adversary repeats the following multiple times (once for each instance): It asks either for the encapsulation of a message comprised of identical blocks, or for the encapsulation of a message consisting of uniformly-generated blocks. The adversary outputs 1 if any two blocks that form the ciphertext collide. If the ciphertext is the encapsulation of the identical-block message then the adversary does not find a collision, since $F(K, \cdot)$ is a permutation for each key $K \in \mathcal{K}$ and is evaluated on distinct input values. Otherwise the ciphertext blocks are random, and one can thus find a collision.

The theorem uses the technical condition that $N\ell(\ell - 1)/|\mathcal{T}| \leq 4$, where ℓ is a parameter that determines the length of the encapsulated messages, measured in blocks. Note that adversaries that could process values N, ℓ that are too large to fulfill this bound will reach at least the same advantage as

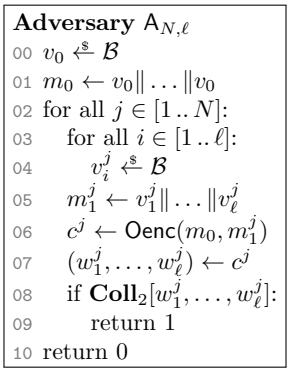

Figure 14: Definition of adversary $\mathsf{A}_{N,\ell}$ against U-MIOT-IND security of CTR+ instantiated with a permutation $F(K, \cdot)$. In line 01 message m_0 is made of ℓ identical blocks.

adversaries considered by the theorem, simply by refraining from posing queries. The stated lower-bound is roughly $N\ell^2/|\mathcal{T}|$ and effectively induced by N applications of the PRP/PRF switching lemma. Note that if the above condition is met with equality, the adversary's advantage is at least 1/2. Further, if $|\mathcal{T}| = |\mathcal{B}| = 2^{128}$, $\ell = 2^{40}$ (this corresponds to a message length of 16 terabytes) and we have $N = 2^{48}$ instances, the success probability of A is about 1/8, or larger.

Theorem 6.3 *Consider* CTR+ *instantiated with a family of permutations $F(K,\cdot)$ over $\mathcal{B}$, and let $N \geq 2$. Assume moreover that $N\ell(\ell-1) \leq 4 \cdot |\mathcal{T}|$. Then for the adversary* A *in Figure 14 it holds:*

$$\mathbf{Adv}^{\mathsf{u\text{-}miot\text{-}ind}}_{\mathsf{CTR+},\mathsf{A},N} \geq \frac{N\ell(\ell-1)}{8 \cdot |\mathcal{T}|} \ .$$

The adversary has a running time of $\mathcal{O}(N\ell \log \ell)$, makes N queries to Oenc *for messages of length at most ℓ and makes no* Odec *queries.*

Proof. We start with the analysis of the running time of A: It is predominantly determined by the search for collisions among ℓ blocks for each of the N iterations of the main loop, which yields a bound of $\mathcal{O}(N\ell \log \ell)$ on the time. To bound $\mathbf{Adv}^{\mathsf{u\text{-}miot\text{-}ind}}_{\mathsf{CTR+},\mathsf{A},N}$ we compute the probability that the adversary outputs 1 depending on the game bit b.

Case $\mathsf{U\text{-}MIOT\text{-}IND}^0$. For each instance $j \in [1\,..\,N]$ the adversary obtains an encapsulation of a sequence of identical blocks. All blocks composing c^j must be distinct, since for each key K, function $F(K,\cdot)$ is a permutation over $\mathcal{B}$. Therefore the output of this game is always 0 and we have $\Pr[\mathsf{U\text{-}MIOT\text{-}IND}^0_{\mathsf{A},N}] = 0$.

Case $\mathsf{U\text{-}MIOT\text{-}IND}^1$. Let p be the probability that there is a collision between ℓ random variables that are uniformly distributed in the set $\mathcal{B}$. We show that for each $j \in [1\,..\,N]$ the probability of A to output 1 when running the j-th iteration of the loop is p. From the definition of Oenc we can write $w_i^j = v_i^j \oplus F(K_j, [\![t_j + i]\!]_L)$ for each $i \in [1\,..\,\ell]$, where K_j and t_j are the key-tag pairs generated by the game $\mathsf{U\text{-}MIOT\text{-}IND}^1_{\mathsf{A},N}$. The elements $v_1^j, \ldots, v_\ell^j$ are generated uniformly in $\mathcal{B}$ and independently of K_j, t_j, their index, and from each other. This means that the elements $w_1^j, \ldots, w_\ell^j$ are also uniformly distributed in $\mathcal{B}$ and mutually independent, even in the presence of colliding keys among $K_1, \ldots, K_N$. Since all blocks v_i^j with $i \in [1\,..\,\ell]$ and $j \in [1\,..\,N]$ are independently random, the probability that the adversary outputs 1 is:

$$\Pr[\mathsf{U\text{-}MIOT\text{-}IND}^1_{\mathsf{A},N}] = 1 - (1-p)^N \ . \tag{13}$$

Since $\ell(\ell-1) \leq 2\,|\mathcal{B}| = 2\,|\mathcal{T}|$ by our hypotheses we can use Lemma A.1 to bound the probability p as $p \geq \ell(\ell-1)/(4 \cdot |\mathcal{B}|)$. Using Lemma A.4 we can bound Equation (13), and since $N\ell(\ell-1) \leq 4\,|\mathcal{T}| = 4\,|\mathcal{B}|$ we can write:

$$\Pr[\mathsf{U\text{-}MIOT\text{-}IND}^1_{\mathsf{A},N}] \geq \min\left\{\frac{1}{2}, \frac{Np}{2}\right\} \geq \frac{N\ell(\ell-1)}{8 \cdot |\mathcal{B}|} = \frac{N\ell(\ell-1)}{8 \cdot |\mathcal{T}|} \ .$$

This proves the bound. □

7 ADEMs secure against active adversaries

In the preceding section we proposed two ADEMs and proved them multi-instance secure against passive adversaries. However, the constructions are based on counter mode encryption and obviously vulnerable in settings with active adversaries that manipulate ciphertexts on the wire. In this section we alleviate the situation by constructing ADEMs that remain secure in the presence of active attacks. Concretely, in line with the encrypt-then-MAC approach [6], we show that an ADEM that is secure against active adversaries can be built from one that is secure against passive adversaries by tamper-protecting its ciphertexts using a message authentication code (MAC). More precisely, with the goal of *tightly* achieving multi-instance security, we use an *augmented message authentication code*[4] (AMAC) where the generation and verification algorithms depend on an auxiliary input: the tag. In the combined construction, the same tag is used for both ADEM and AMAC. As before, using KEM ciphertexts as tags is a reasonable choice. We conclude the section by constructing a (tightly) secure AMAC based on a hash function.

7.1 Augmented message authentication

Augmented message authentication. An augmented message authentication code $\mathsf{AMAC} = (\mathsf{M.mac}, \mathsf{M.vrf})$ for a message space $\mathcal{M}$ is a pair of deterministic algorithms associated with a finite key space $\mathcal{K}$, a tag space $\mathcal{T}$, and a code space $\mathcal{C}$. The algorithm $\mathsf{M.mac}$ takes a key $K \in \mathcal{K}$, a tag $t \in \mathcal{T}$, and a message $m \in \mathcal{M}$, and outputs a code $c \in \mathcal{C}$. The verification algorithm $\mathsf{M.vrf}$ takes a key $K \in \mathcal{K}$, a tag $t \in \mathcal{T}$, a message $m \in \mathcal{M}$, and a code $c \in \mathcal{C}$, and outputs either *true* or *false*. The correctness requirement is that for all $K \in \mathcal{K}$, $t \in \mathcal{T}$, $m \in \mathcal{M}$ and $c \in [\mathsf{M.mac}(K, t, m)]$ we have $\mathsf{M.vrf}(K, t, m, c) = \mathit{true}$.

Augmented message authentication with nonces. We give a game-based authenticity model for AMACs.[11] In our model, for each of a total of N independent keys the adversary can request one MAC code computation but many verifications. The restriction is that for each key the MAC query has to precede all verification queries, and that always the same tag is used. Further, in line with the definition of nonce-based security for ADEMs, we require the tag provided in each MAC computation request to be unique (across all instances). We formalize the corresponding security notion of (strong) nonce-based multi-instance one-time unforgeability for AMACs via the game specified in Figure 15. For a scheme AMAC, to any adversary A and any number of instances N we associate the advantage $\mathbf{Adv}^{\text{n-miot-uf}}_{\mathsf{AMAC},\mathsf{A},N} := \Pr[\mathsf{N\text{-}MIOT\text{-}UF}_{\mathsf{A},N}]$.

Game $\mathsf{N\text{-}MIOT\text{-}UF}_{\mathsf{A},N}$	**Oracle** $\mathsf{Omac}(j, t, m)$	**Oracle** $\mathsf{Ovrf}(j, m, c)$
00 $\mathsf{forged} \leftarrow 0$	07 if $C_j \neq \emptyset$: return $\perp$	13 if $C_j = \emptyset$: return $\perp$
01 $T \leftarrow \emptyset$	08 if $t \in T$: return $\perp$	14 if $(m, c) \in C_j$: return $\perp$
02 for all $j \in [1 \,..\, N]$:	09 $T \leftarrow T \cup \{t\}$; $t_j \leftarrow t$	15 if $\mathsf{M.vrf}(K_j, t_j, m, c)$:
03 $K_j \xleftarrow{\$} \mathcal{K}$	10 $c \leftarrow \mathsf{M.mac}(K_j, t_j, m)$	16 $\mathsf{forged} \leftarrow 1$
04 $C_j \leftarrow \emptyset$	11 $C_j \leftarrow C_j \cup \{(m, c)\}$	17 return *true*
05 run A	12 return c	18 return *false*
06 return forged		

Figure 15: AMAC security game $\mathsf{N\text{-}MIOT\text{-}UF}_{\mathsf{A},N}$, modeling nonce-based multi-instance one-time unforgeability for N instances. Adversary A can access oracles Omac and Ovrf. The tags in line 15 are the same as the ones in line 10.

7.2 The ADEM-then-AMAC construction

Let ADEM and AMAC be an ADEM and an AMAC, respectively. Following the generic encrypt-then-MAC [6] composition technique, and assuming ADEM is secure against passive adversaries, we combine the two schemes to obtain the augmented data-encapsulation mechanism ADEM', which we prove secure against active adversaries. More formally, if $\mathsf{ADEM} = (\mathsf{A.enc}, \mathsf{A.dec})$ and $\mathsf{AMAC} = (\mathsf{M.mac}, \mathsf{M.vrf})$ have key spaces $\mathcal{K}_{\mathsf{dem}}$ and $\mathcal{K}_{\mathsf{mac}}$, respectively, then the key space of ADEM' is $\mathcal{K}_{\mathsf{dem}} \times \mathcal{K}_{\mathsf{mac}}$, and its algorithms are as in Figure 16. Note that the tag space is the same for all three schemes (and that the message spaces have to be sufficiently compatible to each other).

Proc $\mathsf{A.enc}'(K, t, m)$	**Proc** $\mathsf{A.dec}'(K, t, c)$
00 $(K_{\mathsf{dem}}, K_{\mathsf{mac}}) \leftarrow K$	05 $(K_{\mathsf{dem}}, K_{\mathsf{mac}}) \leftarrow K$
01 $c_{\mathsf{dem}} \leftarrow \mathsf{A.enc}(K_{\mathsf{dem}}, t, m)$	06 $(c_{\mathsf{dem}}, c_{\mathsf{mac}}) \leftarrow c$
02 $c_{\mathsf{mac}} \leftarrow \mathsf{M.mac}(K_{\mathsf{mac}}, t, c_{\mathsf{dem}})$	07 if $\mathsf{M.vrf}(K_{\mathsf{mac}}, t, c_{\mathsf{dem}}, c_{\mathsf{mac}})$:
03 $c \leftarrow (c_{\mathsf{dem}}, c_{\mathsf{mac}})$	08 $m \leftarrow \mathsf{A.dec}(K_{\mathsf{dem}}, t, c_{\mathsf{dem}})$
04 return c	09 return m
	10 return $\perp$

Figure 16: Construction of ADEM' from ADEM and AMAC.

[11] In principle we could give two security definitions: one using uniform tags and one using nonce tags. In this paper we formalize only the latter, not the former, for mainly two reasons: (a) the nonce-based notion is not required for our results; (b) in the nonce setting it is not clear how to prove a result similar to the one of Theorem 7.1. The reason for (b) is that to simulate an encapsulation query for a $\mathsf{U\text{-}MIOT\text{-}IND}$ adversary using an AMAC oracle one must specify the tag that is also used to generate the DEM ciphertext, but this is only given as an output of the AMAC oracle.

The proof of the following theorem is in Appendix B.8.

Theorem 7.1 *Let* ADEM' *be constructed from* ADEM *and* AMAC *as described. Then for any number of instances* N *and any ADEM adversary* A *that poses at most* Q_d*-many* Odec *queries, there exist an AMAC adversary* B *and an ADEM adversary* C *such that*

$$\mathbf{Adv}^{\mathsf{n\text{-}miot\text{-}ind}}_{\mathsf{ADEM}',\mathsf{A},N} \leq 2\mathbf{Adv}^{\mathsf{n\text{-}miot\text{-}uf}}_{\mathsf{AMAC},\mathsf{B},N} + \mathbf{Adv}^{\mathsf{n\text{-}miot\text{-}ind}}_{\mathsf{ADEM},\mathsf{C},N} .$$

The running time of B *is at most that of* A *plus the time required to run* N*-many* ADEM *encapsulations and* Q_d*-many* ADEM *decapsulations. The running time of* C *is the same as the running time of* A*. Moreover,* B *poses at most* Q_d*-many* Ovrf *queries, and* C *poses no* Odec *query.*

7.3 A multi-instance secure AMAC

A random oracle directly implies a multi-instance secure AMAC, with a straight-forward construction: the MAC code of a message is computed by concatenating the key, the tag, and the message, and hashing the result. We formalize this as follows. Let $\mathcal{T}$ be a tag space and $\mathcal{M}$ a message space. Let $\mathcal{K}$ and $\mathcal{C}$ be arbitrary finite sets. Let $H\colon \mathcal{K} \times \mathcal{T} \times \mathcal{M} \to \mathcal{C}$ be a hash function. Define function $\mathsf{M.mac}$ and a predicate $\mathsf{M.vrf}$ such that for all K, t, m, c we have $\mathsf{M.mac}(K, t, m) = H(K, t, m)$, and $\mathsf{M.vrf}(K, t, m, c) = \mathit{true}$ iff $H(K, t, m) = c$. Let finally $\mathsf{AMAC} = (\mathsf{M.mac}, \mathsf{M.vrf})$.

Note that hash functions based on the Merkle–Damgård design, like SHA256, do not serve directly as random oracles due to generic length-extension attacks [10], and indeed the ADEM' scheme from Figure 16 is not secure if its AMAC component is derived from such a function. Fortunately, Merkle–Damgård hashing can be modified to achieve indifferentiability from a random oracle [10]. Further, more recent hash functions like SHA3 are naturally resilient against length-extension attacks.

The proof of the following theorem is in Appendix B.9.

Theorem 7.2 *Let* $\mathcal{K}, \mathcal{T}, \mathcal{M}, \mathcal{C}$ *and* $\mathsf{AMAC} = (\mathsf{M.mac}, \mathsf{M.vrf})$ *be as above. If* H *behaves like a (non-programmable) random oracle, for any number of instances* N *and any adversary* A *we obtain*

$$\mathbf{Adv}^{\mathsf{n\text{-}miot\text{-}uf}}_{\mathsf{AMAC},\mathsf{A},N} \leq \frac{q}{|\mathcal{K}|} + \left(\frac{1}{|\mathcal{K}|} + \frac{1}{|\mathcal{C}|}\right) Q_v ,$$

where q *is the number of direct calls to the random oracle by the adversary, and* Q_v *is the number of calls to the oracle* Ovrf*. Note that the bound does not depend on the number of* Omac *queries.*

References

[1] Attrapadung, N., Hanaoka, G., Yamada, S.: A framework for identity-based encryption with almost tight security. In: Iwata, T., Cheon, J.H. (eds.) Advances in Cryptology – ASIACRYPT 2015, Part I. Lecture Notes in Computer Science, vol. 9452, pp. 521–549. Springer, Heidelberg, Germany, Auckland, New Zealand (Nov 30 – Dec 3, 2015) (Cited on page 2.)

[2] Bellare, M.: New proofs for NMAC and HMAC: Security without collision resistance. Journal of Cryptology 28(4), 844–878 (Oct 2015) (Cited on page 13.)

[3] Bellare, M., Bernstein, D.J., Tessaro, S.: Hash-function based PRFs: AMAC and its multi-user security. In: Fischlin, M., Coron, J.S. (eds.) Advances in Cryptology – EUROCRYPT 2016, Part I. Lecture Notes in Computer Science, vol. 9665, pp. 566–595. Springer, Heidelberg, Germany, Vienna, Austria (May 8–12, 2016) (Cited on page 3.)

[4] Bellare, M., Boldyreva, A., Micali, S.: Public-key encryption in a multi-user setting: Security proofs and improvements. In: Preneel, B. (ed.) Advances in Cryptology – EUROCRYPT 2000. Lecture Notes in Computer Science, vol. 1807, pp. 259–274. Springer, Heidelberg, Germany, Bruges, Belgium (May 14–18, 2000) (Cited on page 2, 4, 5, 25.)

[5] Bellare, M., Kilian, J., Rogaway, P.: The security of cipher block chaining. In: Desmedt, Y. (ed.) Advances in Cryptology – CRYPTO'94. Lecture Notes in Computer Science, vol. 839, pp. 341–358. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 21–25, 1994) (Cited on page 13, 18.)

[6] Bellare, M., Namprempre, C.: Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. In: Okamoto, T. (ed.) Advances in Cryptology – ASIACRYPT 2000. Lecture Notes in Computer Science, vol. 1976, pp. 531–545. Springer, Heidelberg, Germany, Kyoto, Japan (Dec 3–7, 2000) (Cited on page 19, 20.)

[7] Bellare, M., Tackmann, B.: The multi-user security of authenticated encryption: AES-GCM in TLS 1.3. In: Robshaw, M., Katz, J. (eds.) Advances in Cryptology – CRYPTO 2016, Part I. Lecture Notes in Computer Science, vol. 9814, pp. 247–276. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 14–18, 2016) (Cited on page 2, 3.)

[8] Chatterjee, S., Koblitz, N., Menezes, A., Sarkar, P.: Another look at tightness II: Practical issues in cryptography. Cryptology ePrint Archive, Report 2016/360 (2016), http://eprint.iacr.org/2016/360 (Cited on page 2, 3, 8.)

[9] Cogliani, S., Maimuţ, D.S., Naccache, D., do Canto, R.P., Reyhanitabar, R., Vaudenay, S., Vizár, D.: OMD: A compression function mode of operation for authenticated encryption. In: Joux, A., Youssef, A.M. (eds.) SAC 2014: 21st Annual International Workshop on Selected Areas in Cryptography. Lecture Notes in Computer Science, vol. 8781, pp. 112–128. Springer, Heidelberg, Germany, Montreal, QC, Canada (Aug 14–15, 2014) (Cited on page 13.)

[10] Coron, J.S., Dodis, Y., Malinaud, C., Puniya, P.: Merkle-Damgård revisited: How to construct a hash function. In: Shoup, V. (ed.) Advances in Cryptology – CRYPTO 2005. Lecture Notes in Computer Science, vol. 3621, pp. 430–448. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 14–18, 2005) (Cited on page 21.)

[11] Cramer, R., Shoup, V.: Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. SIAM Journal on Computing 33(1), 167–226 (2003) (Cited on page 1, 2, 6.)

[12] Gay, R., Hofheinz, D., Kiltz, E., Wee, H.: Tightly CCA-secure encryption without pairings. In: Fischlin, M., Coron, J.S. (eds.) Advances in Cryptology – EUROCRYPT 2016, Part I. Lecture Notes in Computer Science, vol. 9665, pp. 1–27. Springer, Heidelberg, Germany, Vienna, Austria (May 8–12, 2016) (Cited on page 2.)

[13] Gazi, P., Pietrzak, K., Tessaro, S.: Generic security of NMAC and HMAC with input whitening. In: Iwata, T., Cheon, J.H. (eds.) Advances in Cryptology – ASIACRYPT 2015, Part II. Lecture Notes in Computer Science, vol. 9453, pp. 85–109. Springer, Heidelberg, Germany, Auckland, New Zealand (Nov 30 – Dec 3, 2015) (Cited on page 13.)

[14] Gong, J., Chen, J., Dong, X., Cao, Z., Tang, S.: Extended nested dual system groups, revisited. In: Cheng, C.M., Chung, K.M., Persiano, G., Yang, B.Y. (eds.) PKC 2016: 19th International Conference on Theory and Practice of Public Key Cryptography, Part I. Lecture Notes in Computer Science, vol. 9614, pp. 133–163. Springer, Heidelberg, Germany, Taipei, Taiwan (Mar 6–9, 2016) (Cited on page 2.)

[15] Herranz, J., Hofheinz, D., Kiltz, E.: Some (in)sufficient conditions for secure hybrid encryption. Inf. Comput. 208(11), 1243–1257 (2010), http://dx.doi.org/10.1016/j.ic.2010.07.002 (Cited on page 1, 2.)

[16] Hoang, V.T., Tessaro, S.: The multi-user security of double encryption. In: Coron, J., Nielsen, J.B. (eds.) Advances in Cryptology – EUROCRYPT 2017, Part II. Lecture Notes in Computer Science, vol. 10211, pp. 381–411. Springer, Heidelberg, Germany, Paris, France (May 8–12, 2017) (Cited on page 24.)

[17] Hofheinz, D., Jager, T.: Tightly secure signatures and public-key encryption. In: Safavi-Naini, R., Canetti, R. (eds.) Advances in Cryptology – CRYPTO 2012. Lecture Notes in Computer Science, vol. 7417, pp. 590–607. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 19–23, 2012) (Cited on page 2.)

[18] Hofheinz, D., Kiltz, E.: Secure hybrid encryption from weakened key encapsulation. In: Menezes, A. (ed.) Advances in Cryptology – CRYPTO 2007. Lecture Notes in Computer Science, vol. 4622, pp. 553–571. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 19–23, 2007) (Cited on page 1.)

[19] Libert, B., Joye, M., Yung, M., Peters, T.: Concise multi-challenge CCA-secure encryption and signatures with almost tight security. In: Sarkar, P., Iwata, T. (eds.) Advances in Cryptology – ASIACRYPT 2014, Part II. Lecture Notes in Computer Science, vol. 8874, pp. 1–21. Springer, Heidelberg, Germany, Kaoshiung, Taiwan, R.O.C. (Dec 7–11, 2014) (Cited on page 2.)

[20] Libert, B., Peters, T., Joye, M., Yung, M.: Compactly hiding linear spans - tightly secure constant-size simulation-sound QA-NIZK proofs and applications. In: Iwata, T., Cheon, J.H. (eds.) Advances in Cryptology – ASIACRYPT 2015, Part I. Lecture Notes in Computer Science, vol. 9452, pp. 681–707. Springer, Heidelberg, Germany, Auckland, New Zealand (Nov 30 – Dec 3, 2015) (Cited on page 2.)

[21] Patarin, J.: Security in $O(2^n)$ for the xor of two random permutations—proof with the standard H technique. Cryptology ePrint Archive, Report 2013/368 (2013), http://eprint.iacr.org/2013/368 (Cited on page 13.)

[22] Seo, J.H.: Short signatures from Diffie-Hellman: Realizing short public key. Cryptology ePrint Archive, Report 2012/480 (2012), http://eprint.iacr.org/2012/480 (Cited on page 23, 24.)

[23] Shoup, V.: A Computational Introduction to Number Theory and Algebra. Cambridge University Press (2005) (Cited on page 23.)

[24] Zaverucha, G.: Hybrid encryption in the multi-user setting. Cryptology ePrint Archive, Report 2012/159 (2012), http://eprint.iacr.org/2012/159 (Cited on page 2, 3, 8, 11.)

Acknowledgments

We are grateful to Krzysztof Pietrzak and the anonymous reviewers for their valuable comments. The authors were partially supported by ERC Project ERCC (FP7/615074) and by DFG SPP 1736 Big Data.

A Collision probabilities

We give some technical lemmas on collision probabilities. The first two lemmas give bounds on the probability that n random variables collide. The third lemma establishes an upper bound on the collision probability of sets of the form $[\![t \twoheadrightarrow l]\!]_L$ for different values of t. The fourth lemma bounds the expression $1-(1-p)^n$.

Lemma A.1 *Let A be a finite non-empty set of size a. For any $n \in \mathbb{N}$ such that $n(n-1) \leq 2a$ let $X_1, \ldots, X_n$ be independent uniformly distributed random variables over the set A. Then:*

$$\frac{n(n-1)}{4a} \leq \Pr[\mathbf{Coll}_2[X_1, \ldots, X_n]] \leq \frac{n(n-1)}{2a} .$$

Proof. If $n = 0$ or $n = 1$ the bound is immediate. For the remaining cases we invoke the lower bound of [23, Theorem 8.28], namely $\Pr[\mathbf{Coll}_2[X_1, \ldots, X_n]] \geq 1 - e^{-\frac{n(n-1)}{2a}}$. Combining this with the property that $0 < x \leq 1 \Rightarrow 1 - e^{-x} > x/2$ proves the first inequality. The second inequality follows from [23, Theorem 8.26]. □

Lemma A.2 *Let A be a finite non-empty set of size a, let n, m be positive integers, and let $X_1, \ldots, X_n$ be independent uniformly distributed random variables over the set A. Then:*

$$\Pr[\mathbf{Coll}_m[X_1, \ldots, X_n]] \leq \frac{n^m}{m! a^{m-1}} .$$

Proof. This lemma follows immediately from [22, Lemma 2], with $i = j = k = 1$. □

Lemma A.3 *Let $\mathcal{L}$ be a set of size L and l, n, m be positive integers. Suppose that $t_1, \ldots, t_n$ are independent uniformly distributed random variables on the set $\mathbb{Z}/L\mathbb{Z}$. For each $j \in [1\,..\,n]$ we define $T_j = [\![t_j \twoheadrightarrow l]\!]_L$. Then:*

$$\Pr[\exists i_1 \neq \ldots \neq i_m \in [1\,..\,n] : T_{i_1} \cap \ldots \cap T_{i_m} \neq \emptyset] \leq \frac{n^m l^{m-1}}{L^{m-1}} \,.$$

Proof. The proof is a modified version of that of [22, Lemma 2].

The proof consists in a counting argument to upper bound the size of the set S that contains all vectors $t = (t_1, \ldots, t_n) \in \mathbb{Z}/L\mathbb{Z}^n$ for which there exists a non-empty intersection $\Delta_m = T_{i_1} \cap \ldots \cap T_{i_m}$ for some $i_1, \ldots, i_m \in [1\,..\,n]$. The probability that we want to bound is the probability that a uniformly-chosen element of $\mathbb{Z}/L\mathbb{Z}^n$ belongs to S. For any choice of $i_1, \ldots, i_m$, if the set Δ_m is not empty there must exist some index $j \in \{i_1, \ldots, i_m\}$ for which the values at the beginning of all ranges $T_{i_1}, \ldots, T_{i_m}$, i.e., $[\![t_{i_1} + 1]\!]_L, \ldots, [\![t_{i_m} + 1]\!]_L$, belong to T_j.

We can define a vector t as follows: We choose an element $y \in \mathbb{Z}/L\mathbb{Z}$, m distinct values $j, j_1, \ldots, j_{m-1} \in [1\,..\,n]$ and $m-1$ values $y_1, \ldots, y_{m-1}$ such that $[\![y_1 + 1]\!]_L, \ldots, [\![y_{m-1} + 1]\!]_L \in [\![y \twoheadrightarrow l]\!]_L$. The vector t is defined as $t_j = y$, $t_{j_i} = y_i$ for each $i \in [1\,..\,m-1]$, and all remaining elements t_i for each $i \in [1\,..\,n] \setminus \{j, j_1, \ldots, j_{m-1}\}$ are defined arbitrarily in $\mathbb{Z}/L\mathbb{Z}$. The previous description captures all elements in the set S, albeit with repetitions. We can count the total amount of elements that are defined by this procedure as $L \cdot n!/(n-m)! \cdot l^{m-1} \cdot L^{n-m}$.

Since there are in total L^n possible choices for the vector $t \in \mathbb{Z}/L\mathbb{Z}^n$, the probability of a non-empty intersection is upper-bounded by:

$$\frac{L^{n-m+1} l^{m-1} n!}{L^n (n-m)!} \leq \frac{n^m l^{m-1}}{L^{m-1}} \,. \qquad \square$$

Lemma A.4 *Let n be a positive integer. If $0 \leq p \leq 1/n$ then:*

$$\frac{np}{2} \leq 1 - (1-p)^n \leq np \,.$$

Moreover if $1/n < p \leq 1$ then $1 - (1-p)^n > 1/2$.

Proof. The inequality on the left is [16, Lemma 11]. We prove the upper bound by induction on n for any $0 \leq p \leq 1$. The statement holds for $n = 1$. Suppose that $1 - (1-p)^n \leq np$. Then:

$$1 - (1-p)^{n+1} = 1 - (1-p)^n + (1-p)^n p \leq np + p = (n+1)p \,.$$

Suppose now that that $1 \geq p \geq 1/n$. Then we have $1 - (1-p)^n \geq 1 - (1 - 1/n)^n \geq 1 - e^{-1} > 1/2$. $\square$

Lemma A.5 *Let A be a finite non-empty set of size a. For any $n \in \mathbb{N}$ let $X_1, \ldots, X_n$ be independent uniformly distributed random variables over the set A. For every index $j \in [1\,..\,n]$ we define the random variable bad_j as:*

$$\mathrm{bad}_j \coloneqq \begin{cases} 1, & \text{if } \exists k < j \text{ s.t. } X_k = X_j; \\ 0, & \text{otherwise.} \end{cases}$$

We say that the index j is bad *if $\mathrm{bad}_j = 1$. We define the random variable m with values in the set $[0\,..\,n-1]$ as the number of bad indices: $m = \sum_{j=1}^{n} \mathrm{bad}_j$. Then, for any $i \in [0\,..\,n]$:*

$$\Pr[m \geq i] \leq \left(\frac{n^2}{2a}\right)^i \,.$$

Proof. In the following we assume $\Pr[m \geq i] \neq 0$, otherwise the proof is trivial. For all $i \geq 0$, $\Pr[m \geq i+1] = \Pr[m \geq i+1 \mid m \geq i] \cdot \Pr[m \geq i]$. We claim

$$\Pr[m \geq i+1 \mid m \geq i] \leq \frac{n^2}{2a} \,. \tag{14}$$

Then our statement follows together with $\Pr[m \geq 0] \leq 1$.

To prove (14), assume event $m \geq i$ has happened and let h_i be the random variable representing the i-th bad index (and taking value $\perp$ if there is no such index). In the following we condition our probability space with respect to the event $h_i = h$ for some $h \in [i+1\,..\,n]$. Note that $\Pr[h_i = h] \neq 0$ for all $h \in [i+1\,..\,n]$. If we take any configuration of the first h variables for which $h_i = h$, then the remaining $n-h$ can be arbitrarily assigned to any value. This means that one can consider the probability space as fixing the variables $X_1, \ldots, X_h$ according to some distribution and picking the remaining independent variables $X_{h+1}, \ldots, X_n$ uniformly at random. Note that event $m \geq i+1$ happens if and only if one of the indices $h+1, \ldots, n$ is bad. We can write:

$$\begin{aligned}\Pr[m \geq i+1 \mid h_i = h] &= \Pr[\exists j \in [h+1\,..\,n] : \mathrm{bad}_j = 1 \mid h_i = h] \\ &\leq \sum_{j=h+1}^{n} \Pr[\mathrm{bad}_j = 1 \mid h_i = h] \\ &\leq \sum_{j=h+1}^{n} \frac{j-i-1}{a} \leq \frac{(n-i)(n-i-1)}{2a} \leq \frac{n^2}{2a}\,.\end{aligned}$$

To conclude our proof we observe that $\Pr[m \geq i] = \sum_{h=i+1}^{n} \Pr[h_i = h]$ and we write:

$$\begin{aligned}\Pr[m \geq i+1 \mid m \geq i] &= \sum_{h=i+1}^{n} \frac{\Pr[h_i = h]}{\Pr[m \geq i]} \Pr[m \geq i+1 \mid h_i = h] \\ &\leq \sum_{h=i+1}^{n} \frac{\Pr[h_i = h]}{\Pr[m \geq i]} \frac{n^2}{2a} \leq \frac{n^2}{2a}\,.\end{aligned}$$

□

B Proofs

B.1 Proof of Lemma 3.1

The following proof is taken from [4]. We only adjusted the notation.

Proof. Let A be an adversary against multi-user multi-challenge indistinguishability for n users that makes at most q_e challenges per user and q_d decryption queries per user. We build an adversary B against $\mathsf{MUC\text{-}IND}^b_{\mathsf{A},1}$, i.e., standard IND-CCA security, that makes at most one challenge and q_d decryption queries such that:

$$\mathbf{Adv}^{\mathsf{muc\text{-}ind}}_{\mathsf{PKE},\mathsf{B},1} = \frac{1}{n \cdot q_e} \mathbf{Adv}^{\mathsf{muc\text{-}ind}}_{\mathsf{PKE},\mathsf{A},n}\,.$$

In Figure 17 we define the games $\mathsf{G}_{\mathsf{A},u,q}$, with $(u,q) \in [0\,..\,n] \times [0\,..\,q_e]$; we define moreover $p_{u,q} \coloneqq \Pr[\mathsf{G}_{\mathsf{A},u,q}]$. Notice that for the adversary A the two games $\mathsf{G}_{\mathsf{A},0,q_e}$ and $\mathsf{MUC\text{-}IND}^0_{\mathsf{A},n}$, as well as $\mathsf{G}_{\mathsf{A},n,q_e}$ and $\mathsf{MUC\text{-}IND}^1_{\mathsf{A},n}$, have the same output distribution, and in particular $\mathbf{Adv}^{\mathsf{muc\text{-}ind}}_{\mathsf{PKE},\mathsf{A},n} = |p_{0,q_e} - p_{n,q_e}|$. Similarly, for any $u \in [1\,..\,n]$ the games $\mathsf{G}_{\mathsf{A},u,0}$ and $\mathsf{G}_{\mathsf{A},u-1,q_e}$ are equivalent.

Game $\mathsf{G}_{\mathsf{A},u,q}$	**Oracle** $\mathsf{Oenc}(j, m_0, m_1)$	**Oracle** $\mathsf{Odec}(j, c)$
00 $i \leftarrow 0$	06 if $j < u$: $c \xleftarrow{\$} \mathsf{P.enc}(pk_j, m_1)$	14 if $c \in C_j$: return $\perp$
01 for all $j \in [1\,..\,n]$:	07 if $j = u$:	15 $m \leftarrow \mathsf{P.dec}(sk_j, c)$
02 $\quad(pk_j, sk_j) \xleftarrow{\$} \mathsf{P.gen}()$	08 $\quad i \leftarrow i + 1$	16 return m
03 $\quad C_j \leftarrow \emptyset$	09 $\quad$ if $i \leq q$: $c \xleftarrow{\$} \mathsf{P.enc}(pk_j, m_1)$	
04 $b' \xleftarrow{\$} \mathsf{A}(pk_1, \ldots, pk_n)$	10 $\quad$ if $i > q$: $c \xleftarrow{\$} \mathsf{P.enc}(pk_j, m_0)$	
05 return b'	11 if $j > u$: $c \xleftarrow{\$} \mathsf{P.enc}(pk_j, m_0)$	
	12 $C_j \leftarrow C_j \cup \{c\}$	
	13 return c	

Figure 17: Definition of games $\mathsf{G}_{\mathsf{A},u,q}$ for the adversary A. Adversary A is given access to oracles Oenc and Odec.

The adversary B is described in Figure 18. First it generates randomly a pair (u, q). Depending on the bit b of game $\mathsf{MUC\text{-}IND}^b_{\mathsf{B},1}$ against which B is playing, adversary A faces different output distributions. Namely:

- If $b = 0$, then A is playing against game $\mathsf{G}_{\mathsf{A},u,q-1}$.
- If $b = 1$, then A is playing against game $\mathsf{G}_{\mathsf{A},u,q}$.

Adversary $\mathsf{B}(pk)$	if A calls $\mathsf{Oenc}(j, m_0, m_1)$	if A calls $\mathsf{Odec}(j, c)$
00 $(u, q) \xleftarrow{\$} [1\,..\,n] \times [1\,..\,q_e]$	09 if $j < u$: $c \xleftarrow{\$} \mathsf{P.enc}(pk_j, m_1)$	18 if $c \in C_j$: return $\perp$
01 $i \leftarrow 0$	10 if $j = u$:	19 if $j = u$: $m \leftarrow \mathsf{Odec}(1, c)$
02 for all $j \in [1\,..\,n] \setminus \{u\}$:	11 $\quad i \leftarrow i + 1$	20 else: $m \leftarrow \mathsf{P.dec}(sk_j, c)$
03 $\quad (pk_j, sk_j) \xleftarrow{\$} \mathsf{P.gen}()$	12 $\quad$ if $i < q$: $c \xleftarrow{\$} \mathsf{P.enc}(pk_j, m_1)$	21 return m
04 $\quad C_j \leftarrow \emptyset$	13 $\quad$ if $i = q$: $c \xleftarrow{\$} \mathsf{Oenc}(1, m_0, m_1)$	
05 $pk_u \leftarrow pk$	14 $\quad$ if $i > q$: $c \xleftarrow{\$} \mathsf{P.enc}(pk_j, m_0)$	
06 $C_u \leftarrow \emptyset$	15 if $j > u$: $c \xleftarrow{\$} \mathsf{P.enc}(pk_j, m_0)$	
07 $b' \xleftarrow{\$} \mathsf{A}(pk_1, \ldots, pk_n)$	16 $C_j \leftarrow C_j \cup \{c\}$	
08 return b'	17 return c	

Figure 18: The definition of the adversary B. Notice that in lines 13 and 19 the adversary calls its own oracle, which is different from the one called by A.

Since the choice of the pair (u, q) is uniform and independent of the rest of the game we can write:

$$\Pr[\mathsf{MUC\text{-}IND}^0_{\mathsf{B},1}] = \frac{1}{nq_e}\sum_{u=1}^{n}\sum_{q=1}^{q_e} p_{u,q-1}\ , \quad \Pr[\mathsf{MUC\text{-}IND}^1_{\mathsf{B},1}] = \frac{1}{nq_e}\sum_{u=1}^{n}\sum_{q=1}^{q_e} p_{u,q}\ .$$

Combining the previous expressions with $\mathsf{G}_{\mathsf{A},u,0} = \mathsf{G}_{\mathsf{A},u-1,q_e}$ we obtain:

$$\begin{aligned}\mathbf{Adv}^{\mathsf{muc\text{-}ind}}_{\mathsf{PKE},\mathsf{B},1} &= |\Pr[\mathsf{MUC\text{-}IND}^0_{\mathsf{B},1}] - \Pr[\mathsf{MUC\text{-}IND}^1_{\mathsf{B},1}]| \\ &= \frac{1}{n \cdot q_e}|p_{0,q_e} - p_{n,q_e}| = \frac{1}{n \cdot q_e}\mathbf{Adv}^{\mathsf{muc\text{-}ind}}_{\mathsf{PKE},\mathsf{A},n}\ .\end{aligned}$$

The time required to run B can be estimated from the code in Figure 18. □

B.2 Proof of Lemma 3.2

The proof of Lemma 3.2 follows the same steps as the proof of Lemma 3.1 in Appendix B.1. The main difference consists in replacing all encryptions of the message m_b with the pair containing the KEM ciphertext and either the corresponding key K^0 if $b = 0$ or a randomly generated key K^1 if $b = 1$.

B.3 Proof of Lemma 3.3

The proof of Lemma 3.3 is similar to the proof of Lemma 3.1 in Appendix B.1.

Proof. Let A be an adversary against multi-instance one-time indistinguishability for N instances that makes at most Q_d decapsulation queries in total. We build an adversary B against $\mathsf{MIOT\text{-}IND}^b_{\mathsf{A},1}$ that makes at most one encapsulation and Q_d decapsulation queries such that:

$$\mathbf{Adv}^{\mathsf{miot\text{-}ind}}_{\mathsf{DEM},\mathsf{B},1} = \frac{1}{N}\mathbf{Adv}^{\mathsf{miot\text{-}ind}}_{\mathsf{DEM},\mathsf{A},N}\ .$$

In Figure 19 we define the games $\mathsf{G}_{\mathsf{A},q}$, with $q \in [0\,..\,N]$; we define moreover $p_q := \Pr[\mathsf{G}_{\mathsf{A},q}]$. Notice that for the adversary A the two games $\mathsf{G}_{\mathsf{A},0}$ and $\mathsf{MIOT\text{-}IND}^0_{\mathsf{A},N}$, as well as $\mathsf{G}_{\mathsf{A},N}$ and $\mathsf{MIOT\text{-}IND}^1_{\mathsf{A},N}$, have the same output distribution, meaning that $\mathbf{Adv}^{\mathsf{miot\text{-}ind}}_{\mathsf{DEM},\mathsf{A},N} = |p_0 - p_N|$.

The adversary B is described in Figure 20. First it generates randomly the value q. Depending on the bit b of game $\mathsf{MIOT\text{-}IND}^b_{\mathsf{A},1}$ against which B is playing, adversary A faces different output distributions. Namely:

Game $\mathsf{G}_{\mathsf{A},q}$	**Oracle** $\mathsf{Oenc}(j, m_0, m_1)$	**Oracle** $\mathsf{Odec}(j, c)$
00 for all $i \in [1 \,..\, N]$:	05 if $j \leq q$: $c \leftarrow \mathsf{D.enc}(K_j, m_1)$	09 if $C_j = \emptyset$: return $\perp$
01 $\quad K_i \xleftarrow{\$} \mathcal{K}$	06 if $j > q$: $c \leftarrow \mathsf{D.enc}(K_j, m_0)$	10 if $c \in C_j$: return $\perp$
02 $\quad C_i \leftarrow \emptyset$	07 $C_j \leftarrow C_j \cup \{c\}$	11 $m \leftarrow \mathsf{D.dec}(K_j, c)$
03 $b' \xleftarrow{\$} \mathsf{A}$	08 return c	12 return m
04 return b'		

Figure 19: Definition of the games $\mathsf{G}_{\mathsf{A},q}$ for an adversary A. Adversary A is given access to oracles Oenc and Odec.

- If $b = 0$, then A is playing against game $\mathsf{G}_{\mathsf{A},q-1}$.
- If $b = 1$, then A is playing against game $\mathsf{G}_{\mathsf{A},q}$.

Adversary B	if A calls $\mathsf{Oenc}(j, m_0, m_1)$	if A calls $\mathsf{Odec}(j, c)$
00 $q \xleftarrow{\$} [1 \,..\, N]$	07 if $j < q$: $c \xleftarrow{\$} \mathsf{D.enc}(K_j, m_1)$	12 if $c \in C_j$: return $\perp$
01 for all $i \in [1 \,..\, N] \setminus \{q\}$:	08 if $j = q$: $c \xleftarrow{\$} \mathsf{Oenc}(1, m_0, m_1)$	13 if $j = q$: $m \leftarrow \mathsf{Odec}(1, c)$
02 $\quad K_i \xleftarrow{\$} \mathcal{K}$	09 if $j > q$: $c \xleftarrow{\$} \mathsf{D.enc}(K_j, m_0)$	14 else: $m \leftarrow \mathsf{D.dec}(K_j, c)$
03 $\quad C_i \leftarrow \emptyset$	10 $C_j \leftarrow C_j \cup \{c\}$	15 return m
04 $C_q \leftarrow \emptyset$	11 return c	
05 $b' \xleftarrow{\$} \mathsf{A}$		
06 return b'		

Figure 20: The definition of the adversary B. Notice that in lines 08 and 13 the adversary calls its own oracle, which is different from the one called by A.

Since the choice of q is uniform and independent of the rest of the game we can write:

$$\Pr[\mathsf{MIOT\text{-}IND}^0_{\mathsf{A},N}] = \frac{1}{N}\sum_{q=1}^{N} p_{q-1} \;, \quad \Pr[\mathsf{MIOT\text{-}IND}^1_{\mathsf{A},N}] = \frac{1}{N}\sum_{q=1}^{N} p_q \;.$$

Combining the previous expressions we obtain:

$$\begin{aligned}\mathbf{Adv}^{\mathsf{miot\text{-}ind}}_{\mathsf{DEM},\mathsf{B},1} &= |\Pr[\mathsf{MIOT\text{-}IND}^0_{\mathsf{A},1}] - \Pr[\mathsf{MIOT\text{-}IND}^1_{\mathsf{A},1}]| \\ &= \frac{1}{N}|p_0 - p_N| = \frac{1}{N}\mathbf{Adv}^{\mathsf{miot\text{-}ind}}_{\mathsf{DEM},\mathsf{A},N} \;.\end{aligned}$$

The time required to run B can be estimated from the code in Figure 20. □

B.4 Proof of Theorem 3.4

Proof. The game sequence is described in Figure 21.

The game hops are as follows. All modifications involve only the encapsulation oracle. In $\mathsf{G}^{1,b}_{\mathsf{A},n}$ we replace the keys generated by the KEM scheme with randomly generated keys in $\mathcal{K}$. Then, in game $\mathsf{G}^{2,b}_{\mathsf{A},n}$ we flip the bit of the encapsulated message. Eventually, in game $\mathsf{G}^{3,b}_{\mathsf{A},n}$, we switch back from random keys to KEM keys.

First we observe that the game $\mathsf{G}^{0,b}_{\mathsf{A},n}$ (resp. $\mathsf{G}^{3,b}_{\mathsf{A},n}$) is equivalent to the game $\mathsf{MUC\text{-}IND}^0_{\mathsf{A},n}$ (resp. $\mathsf{MUC\text{-}IND}^1_{\mathsf{A},n}$). The main difference consists in storing the encapsulation keys in a list key, which is then used to decapsulate. The correctness of KEM ensures that the procedures are equivalent. Thus we can write $\mathbf{Adv}^{\mathsf{muc\text{-}ind}}_{\mathsf{PKE},\mathsf{A},n} = |\Pr[\mathsf{G}^{0,b}_{\mathsf{A},n}] - \Pr[\mathsf{G}^{3,b}_{\mathsf{A},n}]|$.

We now claim that there exist an adversary B such that:

$$|\Pr[\mathsf{G}^{0,b}_{\mathsf{A},n}] - \Pr[\mathsf{G}^{1,b}_{\mathsf{A},n}]| = \mathbf{Adv}^{\mathsf{muc\text{-}ind}}_{\mathsf{KEM},\mathsf{B},n} \;, \quad |\Pr[\mathsf{G}^{2,b}_{\mathsf{A},n}] - \Pr[\mathsf{G}^{3,b}_{\mathsf{A},n}]| = \mathbf{Adv}^{\mathsf{muc\text{-}ind}}_{\mathsf{KEM},\mathsf{B},n} \;. \tag{15}$$

The adversary B is described in Figure 22. We show that its advantage in breaking multi-user multi-challenge indistinguishability of KEM is exactly $|\Pr[\mathsf{G}^{0,b}_{\mathsf{A},n}] - \Pr[\mathsf{G}^{1,b}_{\mathsf{A},n}]|$. The adversary B is correctly

Game $\mathsf{G}^{0,b}_{\mathsf{A},n}$ – **Game** $\mathsf{G}^{3,b}_{\mathsf{A},n}$

00 for all $j \in [1\,..\,n]$:
01 $\quad (pk_j, sk_j) \xleftarrow{\$} \mathsf{K.gen}()$
02 $\quad C_j \leftarrow \emptyset$
03 $b' \xleftarrow{\$} \mathsf{A}(pk_1, \ldots, pk_n)$
04 return b'

Oracle $\mathsf{Oenc}(j, m_0, m_1)$

05 $(K, c_1) \leftarrow \mathsf{K.enc}(pk_j)$
06 $K \xleftarrow{\$} \mathcal{K}$ | G^1
07 $(K, c_1) \leftarrow \mathsf{K.enc}(pk_j)$ | G^3
08 $\mathsf{key}[j, c_1] \leftarrow K$
09 $c_2 \xleftarrow{\$} \mathsf{D.enc}(K, m_0)$
10 $c_2 \xleftarrow{\$} \mathsf{D.enc}(K, m_1)$ | G^2
11 $c \leftarrow \langle c_1, c_2 \rangle$
12 $C_j \leftarrow C_j \cup \{c\}$
13 return c

Oracle $\mathsf{Odec}(j, \langle c_1, c_2 \rangle)$

14 if $\langle c_1, c_2 \rangle \in C_j$: return $\perp$
15 if $\mathsf{key}[j, c_1] \neq \perp$:
16 $\quad K \leftarrow \mathsf{key}[j, c_1]$
17 else:
18 $\quad K \leftarrow \mathsf{K.dec}(sk_j, c_1)$
19 if $K = \perp$: return $\perp$
20 $m \leftarrow \mathsf{D.dec}(K, c_2)$
21 return m

Figure 21: Definition of the security games $\mathsf{G}^{0,b}_{\mathsf{A},n}$, $\mathsf{G}^{1,b}_{\mathsf{A},n}$, $\mathsf{G}^{2,b}_{\mathsf{A},n}$, and $\mathsf{G}^{3,b}_{\mathsf{A},n}$. Notice that the vector key is initialized implicitly as an empty vector. Adversary A is given access to oracles Oenc and Odec.

Adversary $\mathsf{B}(pk_1, \ldots, pk_n)$

00 for all $j \in [1\,..\,n]$:
01 $\quad C_j \leftarrow \emptyset$
02 $b' \xleftarrow{\$} \mathsf{A}(pk_1, \ldots, pk_n)$
03 return b'

if A calls $\mathsf{Oenc}(j, m_0, m_1)$

04 $(K, c_1) \leftarrow \mathsf{Oenc}(j)$
05 $\mathsf{key}[j, c_1] \leftarrow K$
06 $c_2 \xleftarrow{\$} \mathsf{D.enc}(K, m_0)$
07 $c \leftarrow \langle c_1, c_2 \rangle$
08 $C_j \leftarrow C_j \cup \{c\}$
09 return c

if A calls $\mathsf{Odec}(j, \langle c_1, c_2 \rangle)$

10 if $\langle c_1, c_2 \rangle \in C_j$: return $\perp$
11 if $\mathsf{key}[j, c_1] \neq \perp$:
12 $\quad K \leftarrow \mathsf{key}[j, c_1]$
13 else:
14 $\quad K \leftarrow \mathsf{Odec}(j, c_1)$
15 if $K = \perp$: return $\perp$
16 $m \leftarrow \mathsf{D.dec}(K, c_2)$
17 return m

Figure 22: The definition of the adversary B. Notice that the vector key is initialized implicitly as an empty vector.

simulating the games $\mathsf{MUC\text{-}IND}^b_{\mathsf{A},n}$ for A, depending on the bit b: we just need to prove that no oracle call unexpectedly outputs $\perp$. For this we consider the calls to Odec in line 14. By our definition of key, if the condition in line 11 is not satisfied, then the ciphertext has never been queried before.

We count now the number of queries made by B to the oracles. The adversary A makes at most q_e queries per user to the oracle Oenc, which correspond to at most q_e queries per user to the oracle Oenc. Similarly, A makes at most q_d queries per user to the oracle Odec, which correspond to at most q_d queries per user to the oracle Odec. This verifies the restrictions on the number of queries in our statement.

Observe that one can always transform the adversary B in an adversary B'_{A} to distinguish $\mathsf{G}^{2,b}_{\mathsf{A},n}$ and $\mathsf{G}^{3,b}_{\mathsf{A},n}$. The adversary B'_{A} simply runs the adversary B and swaps the messages m_0 and m_1 for each call to Oenc. In this setting the adversary B is playing the games $\mathsf{G}^{0,b}_{\mathsf{A},n}$ or $\mathsf{G}^{1,b}_{\mathsf{A},n}$. Thus, the probability of B'_{A} to distinguish $\mathsf{G}^{2,b}_{\mathsf{A},n}$ and $\mathsf{G}^{3,b}_{\mathsf{A},n}$ is exactly $\mathbf{Adv}^{\mathsf{muc\text{-}ind}}_{\mathsf{KEM},\mathsf{B},n}$. This justifies Equation (15).

Furthermore we claim that there exist an adversary C such that:

$$|\Pr[\mathsf{G}^{1,b}_{\mathsf{A},n}] - \Pr[\mathsf{G}^{2,b}_{\mathsf{A},n}]| = \mathbf{Adv}^{\mathsf{miot\text{-}ind}}_{\mathsf{DEM},\mathsf{C},nq_e} \,. \tag{16}$$

The adversary C is described in Figure 23. The adversary C is correctly simulating the games $\mathsf{MUC\text{-}IND}^b_{\mathsf{A},n}$ for A, depending on the bit b: we just need to prove that no oracle calls unexpectedly outputs $\perp$. Since the counter i increases before each oracle query, the oracle Oenc is called for the same i only once. Moreover C can call Odec in line 15: if the condition in line 14 is satisfied then c_1 has been used during a previous encapsulation. Hence we can assume that c_2 has not been queried before, otherwise C would have returned $\perp$ in line 13.

We count the number of queries made by C to the oracles. The adversary A makes at most q_e queries per user to the oracle Oenc, which correspond to at most nq_e queries to the oracle Oenc. Similarly, A makes at most q_d queries per user to the oracle Odec, which correspond to at most nq_d queries in total to the oracle Odec. This verifies the restrictions on the number of queries in our statement.

Combining Equation (15) and (16) together we obtain our main statement. The time required to run B and C can be estimated from the code in Figure 22 and Figure 23 respectively. □

Adversary C	if A calls $\mathsf{Oenc}(j, m_0, m_1)$	if A calls $\mathsf{Odec}(j, \langle c_1, c_2 \rangle)$
00 $i \leftarrow 0$	06 $i \leftarrow i + 1$	13 if $\langle c_1, c_2 \rangle \in C_j$: return $\perp$
01 for all $j \in [1 \,..\, n]$:	07 $(K, c_1) \leftarrow \mathsf{K.enc}(pk_j)$	14 if $\mathsf{index}[j, c_1] \neq \perp$:
02 $\quad (pk_j, sk_j) \xleftarrow{\$} \mathsf{K.gen}()$	08 $\mathsf{index}[j, c_1] \leftarrow i$	15 $\quad m \leftarrow \mathsf{Odec}(\mathsf{index}[j, c_1], c_2)$
03 $\quad C_j \leftarrow \emptyset$	09 $c_2 \xleftarrow{\$} \mathsf{Oenc}(i, m_0, m_1)$	16 else:
04 $b' \xleftarrow{\$} \mathsf{A}(pk_1, \ldots, pk_n)$	10 $c \leftarrow \langle c_1, c_2 \rangle$	17 $\quad K \leftarrow \mathsf{K.dec}(j, c_1)$
05 return b'	11 $C_j \leftarrow C_j \cup \{c\}$	18 $\quad$ if $K = \perp$: return $\perp$
	12 return c	19 $\quad m \leftarrow \mathsf{D.dec}(K, c_2)$
		20 return m

Figure 23: The definition of the adversary C. Notice that the vector index is initialized implicitly as an empty vector.

B.5 Proof of Theorem 3.5

Proof. The game sequence is described in Figure 24. Game $\mathsf{G}^{0,b}_{\mathsf{A},n}$ is an instantiation of the security game $\mathsf{MUC\text{-}IND}^b_{\mathsf{A},n}$. Game $\mathsf{G}^{1,b}_{\mathsf{A},n}$ is identical except that the game aborts if two KEM ciphertexts for the same user collide.

Game $\mathsf{G}^{0,b}_{\mathsf{A},n}$ – **Game** $\mathsf{G}^{1,b}_{\mathsf{A},n}$	**Oracle** $\mathsf{Oenc}(j, m_0, m_1)$	**Oracle** $\mathsf{Odec}(j, \langle c_1, c_2 \rangle)$
00 for all $j \in [1 \,..\, n]$:	06 $(K, c_1) \leftarrow \mathsf{K.enc}(pk_j)$	14 if $\langle c_1, c_2 \rangle \in C_j$: return $\perp$
01 $\quad (pk_j, sk_j) \xleftarrow{\$} \mathsf{K.gen}()$	07 if $c_1 \in C_{\mathsf{kem},j}$:	15 $\quad K \leftarrow \mathsf{K.dec}(sk_j, c_1)$
02 $\quad C_j \leftarrow \emptyset$	08 $\quad$ bad $\leftarrow$ *true*; abort $\mid \mathsf{G}^1$	16 if $K = \perp$: return $\perp$
03 $\quad C_{\mathsf{kem},j} \leftarrow \emptyset$	09 $C_{\mathsf{kem},j} \leftarrow C_{\mathsf{kem},j} \cup \{c_1\}$	17 $m \leftarrow \mathsf{D.dec}(K, c_1, c_2)$
04 $b' \xleftarrow{\$} \mathsf{A}(pk_1, \ldots, pk_n)$	10 $c_2 \xleftarrow{\$} \mathsf{D.enc}(K, c_1, m_b)$	18 return m
05 return b'	11 $c \leftarrow \langle c_1, c_2 \rangle$	
	12 $C_j \leftarrow C_j \cup \{c\}$	
	13 return c	

Figure 24: Definition of the security games $\mathsf{G}^{0,b}_{\mathsf{A},n}$ and $\mathsf{G}^{1,b}_{\mathsf{A},n}$. Adversary A is given access to oracles Oenc and Odec.

We claim that the advantage of an adversary to distinguish the two games is

$$|\Pr[\mathsf{G}^{0,b}_{\mathsf{A},n}] - \Pr[\mathsf{G}^{1,b}_{\mathsf{A},n}]| \leq \Pr[\text{bad}] \leq \frac{nq_e{}^2}{2\,|\mathcal{K}|}\,. \tag{17}$$

Notice that if two KEM ciphertexts for the same user collide, then the encapsulated key is the same for both. Since the keys are uniform, the probability of a ciphertext collision for a single user is upper bounded by the probability of two keys to collide. By Lemma A.2 this probability is upper bounded by $q_e{}^2/(2\,|\mathcal{K}|)$ Since the keys are generated independently for each of the n users, our claim follows.

Now we show that for any adversary A against the security of DEM there exists an adversary B against the security of PKE such that:

$$\Pr[\mathsf{G}^{1,b}_{\mathsf{B},n}] = \mathbf{Adv}^{\mathsf{miot\text{-}ind}}_{\mathsf{DEM},\mathsf{A},nq_e}\,. \tag{18}$$

Let A be any adversary against multi-instance one-time indistinguishability with advantage $\mathbf{Adv}^{\mathsf{miot\text{-}ind}}_{\mathsf{DEM},\mathsf{A},nq_e}$. We build an adversary B against multi-user multi-challenge security of the hybrid scheme PKE. The attack is in figure Figure 25.

We argue that, from the point of view of the adversary A called by B, if the game does not abort it is playing against the game $\mathsf{MIOT\text{-}IND}^b_{\mathsf{A},nq_e}$ for DEM. We observe that the keys implicitly generated by KEM are uniform in $\mathcal{K}$. Moreover A does not receive any input that is dependent on the randomness used to generate the DEM keys, if not in the form of the keys themselves. Notably, the decryption procedures uses the KEM ciphertext to call the PKE decryption oracle, but the output message depends uniquely on the corresponding KEM key. The decryption query can always be requested, since by our abort condition on colliding ciphertexts all ciphertext components c_1 appear only once for a each user,

Adversary $\mathsf{B}(pk_1, \ldots, pk_n)$
00 $(u, q) \leftarrow (1, 0)$
01 for all $j \in [1 \,..\, nq_e]$:
02 $\quad C_{\mathsf{dem},j} \leftarrow \emptyset$
03 for all $u \in [1 \,..\, n]$
04 $\quad C_{\mathsf{kem},u} \leftarrow \emptyset$
05 $b' \xleftarrow{\$} \mathsf{A}$
06 return b'

if A calls $\mathsf{Oenc}(j, m_0, m_1)$
07 if $C_{\mathsf{dem},j} \neq \emptyset$: return $\perp$
08 $q \leftarrow q + 1$
09 if $q > q_e$: $(u, q) \leftarrow (u + 1, 1)$
10 $\langle c_1, c_2 \rangle \leftarrow \mathsf{Oenc}(u, m_0, m_1)$
11 if $c_1 \in C_{\mathsf{kem},u}$: abort
12 $C_{\mathsf{dem},j} \leftarrow C_{\mathsf{dem},j} \cup \{c_2\}$
13 $C_{\mathsf{kem},u} \leftarrow C_{\mathsf{kem},u} \cup \{c_1\}$
14 $D_{\mathsf{KEM}}[j] \leftarrow (u, c_1)$
15 return c_2

if A calls $\mathsf{Odec}(j, c_2)$
16 if $C_{\mathsf{dem},j} = \emptyset$: return $\perp$
17 if $c_2 \in C_{\mathsf{dem},j}$: return $\perp$
18 $(u, c_1) \leftarrow D_{\mathsf{KEM}}[j]$
19 $m \leftarrow \mathsf{Odec}(u, \langle c_1, c_2 \rangle)$
20 return m

Figure 25: The definition of the adversary B. Notice that the vector D_{KEM} is initialized implicitly as an empty vector.

and thus we never ask for a decryption of a PKE challenge. This does not change in case of a collision of KEM keys/ciphertexts. Since the victory condition of the DEM security game is the same as that for $\mathsf{G}^{1,b}_{\mathsf{A},n}$ we get Equation 18.

Combining Equation (17) and (18) gives the first part of our statement. By construction there are at most q_e encryption calls to each user. The number of decryption queries by B coincides with the number of decryption queries by A. The running time of the two adversaries is roughly equivalent. □

B.6 Proof of Lemma 5.1

Proof. The adversary B uniformly generates the tags $t_1, \ldots, t_N \in \mathcal{T}$. For each query of A the adversary B forwards the input to the game $\mathsf{N\text{-}MIOT\text{-}IND}^b_{\mathsf{A},N}$, $b \in \{0, 1\}$, after including the corresponding tag. If none of the tags $t_1, \ldots, t_N$ collide, then B is correctly simulating the game $\mathsf{U\text{-}MIOT\text{-}IND}^b_{\mathsf{A},N}$, and the advantages of the two adversaries A and B coincide. Moreover, from Lemma A.2 the probability that two tags collide is upper bounded by $N^2/(2\,|\mathcal{T}|)$. These two observations prove our bound. □

B.7 Proof of Lemma 5.3

Proof. The proof follows closely that of Theorem 3.4. The game sequence is described in Figure 26.

Game $\mathsf{G}^{0,b}_{\mathsf{A},n}$ – **Game** $\mathsf{G}^{2,b}_{\mathsf{A},n}$
00 $C_{\mathsf{kem}} \leftarrow \emptyset$
01 for all $j \in [1 \,..\, n]$:
02 $\quad (pk_j, sk_j) \xleftarrow{\$} \mathsf{K.gen}()$
03 $\quad C_j \leftarrow \emptyset$
04 $b' \xleftarrow{\$} \mathsf{A}(pk_1, \ldots, pk_n)$
05 return b'

Oracle $\mathsf{Oenc}(j, m_0, m_1)$
06 $(K, c_1) \leftarrow \mathsf{K.enc}(pk_j)$
07 if $c_1 \in C_{\mathsf{kem}}$:
08 $\quad$ bad $\leftarrow$ *true*; abort $\quad | \mathsf{G}^2$
09 $C_{\mathsf{kem}} \leftarrow C_{\mathsf{kem}} \cup \{c_1\}$
10 $K \xleftarrow{\$} \mathcal{K} \quad | \mathsf{G}^1$
11 $\mathsf{key}[j, c_1] \leftarrow K$
12 $c_2 \xleftarrow{\$} \mathsf{D.enc}(K, c_1, m_b)$
13 $c \leftarrow \langle c_1, c_2 \rangle$
14 $C_j \leftarrow C_j \cup \{c\}$
15 return c

Oracle $\mathsf{Odec}(j, \langle c_1, c_2 \rangle)$
16 if $\langle c_1, c_2 \rangle \in C_j$: return $\perp$
17 if $\mathsf{key}[j, c_1] \neq \perp$:
18 $\quad K \leftarrow \mathsf{key}[j, c_1]$
19 else:
20 $\quad K \leftarrow \mathsf{K.dec}(sk_j, c_1)$
21 if $K = \perp$: return $\perp$
22 $m \leftarrow \mathsf{D.dec}(K, c_1, c_2)$
23 return m

Figure 26: Definition of the security games $\mathsf{G}^{0,b}_{\mathsf{A},n}$, $\mathsf{G}^{1,b}_{\mathsf{A},n}$, and $\mathsf{G}^{2,b}_{\mathsf{A},n}$. Notice that the vector key is initialized implicitly as an empty vector. Adversary A is given access to oracles Oenc and Odec.

The game hops are as follows. All modifications involve only the encapsulation oracle. In $\mathsf{G}^{1,b}_{\mathsf{A},n}$ we replace the keys generated by the KEM scheme with randomly generated keys in $\mathcal{K}$. In game $\mathsf{G}^{2,b}_{\mathsf{A},n}$ we abort if KEM generates the same ciphertext c_1 more than once.

We observe that game $\mathsf{G}^{0,0}_{\mathsf{A},n}$ (resp. $\mathsf{G}^{0,1}_{\mathsf{A},n}$) has the same output distribution of the security game $\mathsf{MUC\text{-}IND}^0_{\mathsf{A},n}$ (resp. $\mathsf{MUC\text{-}IND}^1_{\mathsf{A},n}$). The main difference consists in storing the encapsulation keys in a list key, which is then used to decapsulate. The correctness of KEM ensures that the procedures are equivalent. We can write $\mathbf{Adv}^{\mathsf{muc\text{-}ind}}_{\mathsf{PKE},\mathsf{A},n} = |\Pr[\mathsf{G}^{0,0}_{\mathsf{A},n}] - \Pr[\mathsf{G}^{0,1}_{\mathsf{A},n}]|$.

We now claim that there exist an adversary B such that for any bit b:

$$|\Pr[\mathsf{G}^{0,b}_{\mathsf{A},n}] - \Pr[\mathsf{G}^{1,b}_{\mathsf{A},n}]| = \mathbf{Adv}^{\mathsf{muc\text{-}ind}}_{\mathsf{KEM},\mathsf{B},n} \,. \tag{19}$$

Adversary $\mathsf{B}(pk_1, \ldots, pk_n)$	if A calls $\mathsf{Oenc}(j, m_0, m_1)$	if A calls $\mathsf{Odec}(j, \langle c_1, c_2\rangle)$
00 for all $j \in [1\,..\,n]$: $C_j \leftarrow \emptyset$	03 $(K, c_1) \leftarrow \mathsf{Oenc}(j)$	09 if $\langle c_1, c_2\rangle \in C_j$: return $\perp$
01 $b' \xleftarrow{\$} \mathsf{A}(pk_1, \ldots, pk_n)$	04 $\mathsf{key}[j, c_1] \leftarrow K$	10 if $\mathsf{key}[j, c_1] \neq \perp$:
02 return b'	05 $c_2 \xleftarrow{\$} \mathsf{D.enc}(K, c_1, m_0)$	11 $\quad K \leftarrow \mathsf{key}[j, c_1]$
	06 $c \leftarrow \langle c_1, c_2\rangle$	12 else:
	07 $C_j \leftarrow C_j \cup \{c\}$	13 $\quad K \leftarrow \mathsf{Odec}(j, c_1)$
	08 return c	14 if $K = \perp$: return $\perp$
		15 $m \leftarrow \mathsf{D.dec}(K, c_1, c_2)$
		16 return m

Figure 27: The definition of the adversary B. Notice that the vector key is initialized implicitly as an empty vector.

The adversary B is described in Figure 27. We show that its advantage in breaking multi-user multi-challenge indistinguishability of KEM is exactly $|\Pr[\mathsf{G}^{0,0}_{\mathsf{A},n}] - \Pr[\mathsf{G}^{1,0}_{\mathsf{A},n}]|$. The adversary B is correctly simulating the games $\mathsf{MUC\text{-}IND}^b_{\mathsf{A},n}$ for A, depending on the bit b: we just need to show that B can call Odec in line 13. By our definition of key, if the condition in line 10 is not satisfied, then the ciphertext has never been queried before, hence B can decapsulate.

We count the number of queries to the oracles that are made by B. The adversary A makes at most q_e queries per user to the oracle Oenc, which correspond to at most q_e queries per user to the oracle Oenc. Similarly, A makes at most q_d queries per user to the oracle Odec, which correspond to at most q_d queries per user to the oracle Odec. This verifies the restrictions on the number of queries in our statement.

Observe that one can always transform the adversary B in an adversary B' to distinguish $\mathsf{G}^{0,1}_{\mathsf{A},n}$ and $\mathsf{G}^{1,1}_{\mathsf{A},n}$. The adversary B' simply runs the adversary B and swaps the messages m_0 and m_1 for each call to Oenc. In this setting the adversary B is playing the games $\mathsf{G}^{0,0}_{\mathsf{A},n}$ or $\mathsf{G}^{1,0}_{\mathsf{A},n}$. The probability of B' to distinguish $\mathsf{G}^{0,1}_{\mathsf{A},n}$ and $\mathsf{G}^{1,1}_{\mathsf{A},n}$ is thus exactly $\mathbf{Adv}^{\mathsf{muc\text{-}ind}}_{\mathsf{KEM},\mathsf{B},n}$. This justifies Equation (19).

Next we argue that, for any bit b:

$$|\Pr[\mathsf{G}^{1,b}_{\mathsf{A},n}] - \Pr[\mathsf{G}^{2,b}_{\mathsf{A},n}]| \leq \Pr[\mathrm{bad}] \leq \binom{N}{2} p \,. \tag{20}$$

In fact the probability that the game $\mathsf{G}^{2,b}_{\mathsf{A},n}$ aborts during the i-th encryption query is lower bounded by $(i-1)p$. Summing up for all the $N = nq_e$ queries yields exactly Equation (20).

Finally we claim that there exist an adversary C such that:

$$|\Pr[\mathsf{G}^{2,0}_{\mathsf{A},n}] - \Pr[\mathsf{G}^{2,1}_{\mathsf{A},n}]| = \mathbf{Adv}^{\mathsf{miot\text{-}ind}}_{\mathsf{DEM},\mathsf{C},nq_e} \,. \tag{21}$$

The adversary C is described in Figure 28. The adversary C is correctly simulating the games $\mathsf{MUC\text{-}IND}^b_{\mathsf{A},n}$ for A, depending on the bit b: we just need to show that the adversary can call the oracles without receiving any invalid output $\perp$. Since the counter i is always increasing before each oracle query, the oracle Oenc is called only once for the same i. From our abort condition we also know that the tag used as input to the same oracle has never been queried before. We argue similarly for the oracle Odec in line 18. Suppose that the oracle has been queried with input $(j, \langle c_1, c_2\rangle)$. If the condition in line 17 is satisfied then c_1 was used as tag during the previous encapsulation for the $\mathsf{index}[j, c_1]$-th instance, thus becoming implicitly assigned to that instance for decapsulation. We also see that c_2 has not been queried before, otherwise C would have returned $\perp$ in line 16.

We count the number of queries to the oracles that are made by C. The adversary A makes at most q_e queries per user to the oracle Oenc, which correspond to at most nq_e queries to the oracle Oenc. Similarly, A makes at most q_d queries per user to the oracle Odec, which correspond to at most nq_d queries in total to the oracle Odec. This verifies the restrictions on the number of queries in our statement.

Combining Equation (19) and (20) (both for $b = 0$ and $b = 1$) with Equation (21) we obtain our main statement. The time required to run B and C can be estimated from the code in Figure 27 and Figure 28 respectively. □

Adversary C	if A calls $\mathsf{Oenc}(j, m_0, m_1)$	if A calls $\mathsf{Odec}(j, \langle c_1, c_2 \rangle)$
00 $C_{\mathsf{kem}} \leftarrow \emptyset$	07 $i \leftarrow i + 1$	16 if $\langle c_1, c_2 \rangle \in C_j$: return $\perp$
01 $i \leftarrow 0$	08 if $c_1 \in C_{\mathsf{kem}}$: abort	17 if $\mathsf{index}[j, c_1] \neq \perp$:
02 for all $j \in [1 \,..\, n]$:	09 $C_{\mathsf{kem}} \leftarrow C_{\mathsf{kem}} \cup \{c_1\}$	18 $m \leftarrow \mathsf{Odec}(\mathsf{index}[j, c_1], c_2)$
03 $(pk_j, sk_j) \xleftarrow{\$} \mathsf{K.gen}()$	10 $(K, c_1) \leftarrow \mathsf{K.enc}(pk_j)$	19 else:
04 $C_j \leftarrow \emptyset$	11 $\mathsf{index}[j, c_1] \leftarrow i$	20 $K \leftarrow \mathsf{K.dec}(j, c_1)$
05 $b' \xleftarrow{\$} \mathsf{A}(pk_1, \ldots, pk_n)$	12 $c_2 \xleftarrow{\$} \mathsf{Oenc}(i, c_1, m_0, m_1)$	21 if $K = \perp$: return $\perp$
06 return b'	13 $c \leftarrow \langle c_1, c_2 \rangle$	22 $m \leftarrow \mathsf{D.dec}(K, c_1, c_2)$
	14 $C_j \leftarrow C_j \cup \{c\}$	23 return m
	15 return c	

Figure 28: The definition of the adversary C. Notice that the vector index is initialized implicitly as an empty vector.

B.8 Proof of Lemma 7.1

Proof. The proof involves the two games specified in Figure 29. The game $\mathsf{G}^{0,b}_{\mathsf{A},N}$ coincides with $\mathsf{N\text{-}MIOT\text{-}IND}^{b}_{\mathsf{A},N}$ instantiated with the ADEM of Figure 16. We correspondingly have $\mathbf{Adv}^{\mathsf{n\text{-}miot\text{-}ind}}_{\mathsf{ADEM}',\mathsf{A},N} = |\Pr[\mathsf{G}^{0,0}_{\mathsf{A},N}] - \Pr[\mathsf{G}^{0,1}_{\mathsf{A},N}]|$. Game $\mathsf{G}^{1,b}_{\mathsf{A},N}$ coincides with game $\mathsf{G}^{0,b}_{\mathsf{A},N}$ insofar as the adversary never submits a query to Omac that passes the MAC verification in line 19; otherwise the flag bad is set to *true* and the game aborts.

Game $\mathsf{G}^{0,b}_{\mathsf{A},N}$ – $\mathsf{G}^{1,b}_{\mathsf{A},N}$	**Oracle** $\mathsf{Oenc}(j, t, m_0, m_1)$	**Oracle** $\mathsf{Odec}(j, c)$	
00 $T \leftarrow \emptyset$	06 if $C_j \neq \emptyset$: return $\perp$	15 if $C_j = \emptyset$: return $\perp$	
01 for all $j \in [1 \,..\, N]$:	07 if $t \in T$: return $\perp$	16 if $c \in C_j$: return $\perp$	
02 $K_j \xleftarrow{\$} \mathcal{K}$	08 $T \leftarrow T \cup \{t\}$; $t_j \leftarrow t$	17 $(K_{\mathsf{dem}}, K_{\mathsf{mac}}) \leftarrow K_j$	
03 $C_j \leftarrow \emptyset$	09 $(K_{\mathsf{dem}}, K_{\mathsf{mac}}) \leftarrow K_j$	18 $(c_{\mathsf{dem}}, c_{\mathsf{mac}}) \leftarrow c$	
04 $b' \xleftarrow{\$} \mathsf{A}$	10 $c_{\mathsf{dem}} \leftarrow \mathsf{A.enc}(K_{\mathsf{dem}}, t_j, m_b)$	19 if $\mathsf{M.vrf}(K_{\mathsf{mac}}, t_j, c_{\mathsf{dem}}, c_{\mathsf{mac}})$:	
05 return b'	11 $c_{\mathsf{mac}} \leftarrow \mathsf{M.mac}(K_{\mathsf{mac}}, t_j, c_{\mathsf{dem}})$	20 bad $\leftarrow$ *true*; abort	G^1
	12 $c \leftarrow (c_{\mathsf{dem}}, c_{\mathsf{mac}})$	21 $m \leftarrow \mathsf{A.dec}(K_{\mathsf{dem}}, t_j, c_{\mathsf{dem}})$	
	13 $C_j \leftarrow C_j \cup \{c\}$	22 return m	
	14 return c	23 return $\perp$	

Figure 29: The security games $\mathsf{G}^{0,b}_{\mathsf{A},N}$ and $\mathsf{G}^{1,b}_{\mathsf{A},N}$. Adversary A is given access to oracles Oenc and Odec. When the game "aborts" this means it stops executing and returns 0.

We argue that any adversary A playing the game $\mathsf{G}^{0,b}_{\mathsf{A},N}$ obtaining any output other than $\perp$ from the oracle Odec can be used to break the security of AMAC. Concretely, in Figure 30 we build an adversary B against AMAC that breaks the security game $\mathsf{N\text{-}MIOT\text{-}UF}_{\mathsf{B},N}$ by calling the adversary A and answering its oracle queries using the oracles Omac and Ovrf.

Adversary B	if A calls $\mathsf{Oenc}(j, t, m_0, m_1)$	if A calls $\mathsf{Odec}(j, c)$
00 $T \leftarrow \emptyset$	06 if $C_j \neq \emptyset$: return $\perp$	14 if $C_j = \emptyset$: return $\perp$
01 for all $j \in [1 \,..\, N]$:	07 if $t \in T$: return $\perp$	15 if $c \in C_j$: return $\perp$
02 $K_j \xleftarrow{\$} \mathcal{K}_{\mathsf{dem}}$	08 $T \leftarrow T \cup \{t\}$; $t_j \leftarrow t$	16 $(c_{\mathsf{dem}}, c_{\mathsf{mac}}) \leftarrow c$
03 $C_j \leftarrow \emptyset$	09 $c_{\mathsf{dem}} \leftarrow \mathsf{A.enc}(K_j, t_j, m_b)$	17 $\mathsf{forged} \leftarrow \mathsf{Ovrf}(j, c_{\mathsf{dem}}, c_{\mathsf{mac}})$
04 run A	10 $c_{\mathsf{mac}} \leftarrow \mathsf{Omac}(j, t_j, c_{\mathsf{dem}})$	18 if $\neg\,\mathsf{forged}$: return $\perp$
05 return $\perp$	11 $c \leftarrow (c_{\mathsf{dem}}, c_{\mathsf{mac}})$	19 $m \leftarrow \mathsf{A.dec}(K_j, t_j, c_{\mathsf{dem}})$
	12 $C_j \leftarrow C_j \cup \{c\}$	20 return m
	13 return c	

Figure 30: Adversary B against game $\mathsf{N\text{-}MIOT\text{-}UF}$. It has access to oracles Omac and Ovrf.

The probability that the condition in line 19 of Figure 29 is triggered is exactly the probability that B

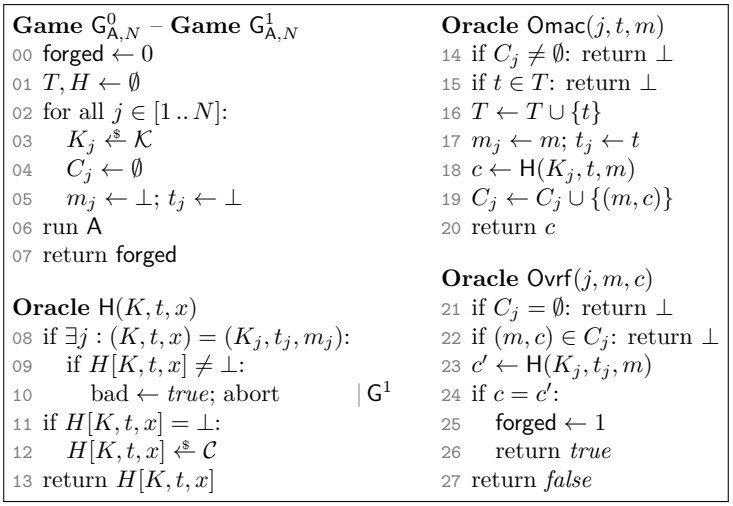

Figure 31: The security games $\mathsf{G}^0_{\mathsf{A},N}$ and $\mathsf{G}^1_{\mathsf{A},N}$. Adversary A is given access to oracles H, Omac, and Ovrf. When the game "aborts" this means it stops executing and returns 0.

wins the game $\mathsf{N\text{-}MIOT\text{-}UF}_{\mathsf{B},N}$ for the scheme AMAC, hence, for any bit $b \in \{0, 1\}$:

$$|\Pr[\mathsf{G}^{0,b}_{\mathsf{A},N}] - \Pr[\mathsf{G}^{1,b}_{\mathsf{A},N}]| \leq \Pr[\text{bad}] = \mathbf{Adv}^{\mathsf{n\text{-}miot\text{-}uf}}_{\mathsf{AMAC},\mathsf{B},N} .$$

Finally, game $\mathsf{G}^{1,b}_{\mathsf{A},N}$ is equivalent to the game $\mathsf{N\text{-}MIOT\text{-}IND}_{\mathsf{A},N}$ with no calls to the decapsulation oracle. Hence if we define the adversary C that runs A forwarding all calls to the oracle Oenc and answering all calls to Odec with $\perp$ we obtain:

$$|\Pr[\mathsf{G}^{1,0}_{\mathsf{A},N}] - \Pr[\mathsf{G}^{1,1}_{\mathsf{A},N}]| = \mathbf{Adv}^{\mathsf{n\text{-}miot\text{-}ind}}_{\mathsf{ADEM},\mathsf{C},N} .$$

Combining the previous formulas for both $b \in \{0, 1\}$ yields our statement. □

B.9 Proof of Lemma 7.2

Proof. The proof involves the two games specified in Figure 31. The game $\mathsf{G}^0_{\mathsf{A},N}$ is the game $\mathsf{N\text{-}MIOT\text{-}UF}_{\mathsf{A},N}$ instantiated with the described hash-based AMAC, with the random oracle made explicit. We correspondingly have $\mathbf{Adv}^{\mathsf{n\text{-}miot\text{-}uf}}_{\mathsf{AMAC},\mathsf{A},N} = \Pr[\mathsf{G}^0_{\mathsf{A},N}]$. Game $\mathsf{G}^1_{\mathsf{A},N}$ is equivalent to game $\mathsf{G}^0_{\mathsf{A},N}$ except that the adversary never queries the random oracle for any input that was used to generate any output of Omac.[12]

We claim that $|\Pr[\mathsf{G}^0_{\mathsf{A},N}] - \Pr[\mathsf{G}^1_{\mathsf{A},N}]| \leq \Pr[\text{bad}] \leq q/|\mathcal{K}|$. To show this, we analyze the abort condition in line 10. Notice that the adversary has no information on the MAC keys but full control on the inputs to Omac. The probability of the adversary to trigger the abort condition stems solely from key guessing. Since all tags $t_1, \ldots, t_N$ are used at most once for each call to Omac, the adversary guesses at most one key at a time for each call to the random oracle. The probability to guess the key on a single query to H is at most $1/|\mathcal{K}|$. Repeating this argument for each of the q queries to H yields our claim.

From game $\mathsf{G}^1_{\mathsf{A},N}$ the adversary has no additional information on the MAC keys, in particular it cannot determine whether any value in $\mathcal{C}$ is a valid code for a fixed instance $j \in [1\,..\,N]$ and message $m \in \mathcal{M}$. It remains to compute the probability that the adversary is able to guess the code by chance while calling Ovrf, which coincides with $\Pr[\mathsf{G}^1_{\mathsf{A},N}]$. This is achieved either by guessing the key K_j and sending to the oracle the correct input $(j, m, \mathsf{H}(K_j, t_j, m))$, or by guessing the correct code in $\mathcal{C}$. This yields:

$$\Pr[\mathsf{G}^1_{\mathsf{A},N}] \leq \left(\frac{1}{|\mathcal{K}|} + \frac{1}{|\mathcal{C}|}\right) Q_v . \tag{22}$$

Applying the triangle inequality to our previous equations gives our statement. □

[12] Notice that our abort condition consider both the case in which the adversary queries first Omac and later guesses the inputs to H that were used by the reduction to answer the query, and that in which the order of the two queries is switched.