The Committing Security of MACs with Applications to Generic Composition*

Ritam Bhaumik[1] Bishwajit Chakraborty[2] Wonseok Choi[3] Avijit Dutta[4] Jérôme Govinden[5] Yaobin Shen[6]

[1] EPFL, Switzerland
ritam.bhaumik@epfl.ch
[2] Nanyang Technological University, Singapore
bishwajit.chakrabort@ntu.edu.sg
[3] Purdue University, West Lafayette, IN, USA
wonseok@purdue.edu
[4] Institute for Advancing Intelligence, TCG CREST, India
avijit.dutta@tcgcrest.org
[5] Technische Universität Darmstadt, Germany
jerome.govinden@tu-darmstadt.de
[6] School of Informatics, Xiamen University, Xiamen, China
yaobin.shen@xmu.edu.cn

Abstract

Message Authentication Codes (MACs) are ubiquitous primitives deployed in multiple flavors through standards such as HMAC, CMAC, GMAC, LightMAC, and many others. Its versatility makes it an essential building block in applications necessitating message authentication and integrity checks, in authentication protocols, authenticated encryption schemes, or as a pseudorandom or key derivation function. Its usage in this variety of settings makes it susceptible to a broad range of attack scenarios. The latest attack trends leverage a lack of commitment or context-discovery security in AEAD schemes and these attacks are mainly due to the weakness in the underlying MAC part. However, these new attack models have been scarcely analyzed for MACs themselves. This paper provides a thorough treatment of MACs committing and context-discovery security. We reveal that commitment and context-discovery security of MACs have their own interest by highlighting real-world vulnerable scenarios. We formalize the required security notions for MACs, and analyze the security of standardized MACs for these notions. Additionally, as a constructive application, we analyze generic AEAD composition and provide simple and efficient ways to build committing and context-discovery secure AEADs.

1 Introduction

Initially conceived to address the limitations of privacy-only schemes, authenticated encryption (AE) was defined to provide authentication and confidentiality simultaneously. It is then transitioned from probabilistic to deterministic nonce-based algorithms to prevent attacks leveraging the use of weak randomness. In its contemporary form, called authenticated encryption with associated data (AEAD), it also authenticates an additional associated data alongside the message. Its encryption procedure $\mathsf{AE.Enc}$ takes as input

*© IACR 2024. This is the full version of the work that appeared at CRYPTO 2024.

a key K, a nonce N, an associated data A and a message M to return a ciphertext $C \leftarrow \mathsf{AE.Enc}(K, N, A, M)$. Widely deployed standards such as AES-GCM [Dwo07a] and ChaCha20-Poly1305 [NL18] exemplify the prevalence of such AEAD schemes in modern applications. These standards undergo rigorous scrutiny and are typically accompanied by formal security proofs [HTT18, DGGP21] to instill confidence in their cryptographic properties.

Committing Security. Despite this apparent guarantee, a recent line of research has found a variety of settings vulnerable to attacks exploiting the lack of key commitment in popular AEAD schemes [DGRW18, ADG+22, LGR21]. In essence, these attacks exploit the fact that an adversary can find a pair $(K_1, N_1, A_1, M_1), (K_2, N_2, A_2, M_2)$ satisfying $K_1 \neq K_2$ such that $\mathsf{AE.Enc}(K_1, N_1, A_1, M_1) = \mathsf{AE.Enc}(K_2, N_2, A_2, M_2)$. These exploits were possible only because they lie outside of the classical security model for AEAD. To prevent these newfound vulnerabilities, new security notions were defined for AEAD, starting with *key-committing* security [ADG+22], which states that it is hard to find the previously described colliding pair satisfying $K_1 \neq K_2$. Subsequently, this notion evolved into the stronger notion of *context-committing* security [BH22] requiring instead to find a colliding pair satisfying $(K_1, N_1, A_1) \neq (K_2, N_2, A_2)$.

Yet until now, real-world attacks have only predominantly exploited the lack of key-committing security in AEAD, raising the question of whether key-committing security should be the aimed security goal instead of the stronger context-committing notion. Noteworthy is the practical advantage of key-committing security, which can be efficiently obtained with minimal modifications to existing schemes [BH22, ADG+22]. Conversely, the stronger context-committing notion necessitates the integration of an additional primitive—a costly hash function—applied over the key, nonce, and associated data alongside the base AEAD scheme [BH22, CR22]. Menda et al. [MLGR23] introduced an orthogonal security notion called *context-discovery* security (CDY) that is to preimage resistance as context-committing security is to collision resistance. However, no real-world application has yet been found for this new security notion, and no AEAD has been proven context-discovery secure. With this abundance of new security notions, it is of interest to identify a definition that strikes a balance between simplicity for security proofs and versatility for widespread application.

This growing list of vulnerable settings to key-committing attacks has led to a call from part of the cryptographic community to standardize a context-committing AEAD scheme [BCG+, BHW, MST], driven by the absence of any standardized context-committing secure schemes. While a few fixes have been proposed [BH22, CR22], several duplex-based AEADs have recently been proven context-committing secure [DFG23, DMVA23, KSW23, NSS23]. However, the current landscape reveals a notable gap: existing duplex designs primarily target lightweight applications, and thus no dedicated constructions tailored for general-purpose and high-performance scenarios exist. Furthermore, compared to prevailing standards, proposed fixes [BH22, CR22] introduce overheads such as having longer ciphertext expansion, necessitating two passes over the associated data, or needing an additional primitive. Ideally, the quest for an efficient construction would necessitate leveraging existing implementations while minimizing the reliance on additional primitives.

MAC-based AEADs. A message authentication code (MAC) is an older and simpler primitive than AEAD, aiming only at authenticating data between parties that share a key. It is also widely deployed through standards such as HMAC [KBC97] and CMAC [Dwo05]. Beyond its traditional use, it has been adopted for a variety of applications, such as for a pseudorandom function (PRF), a swap-PRF, a key combiner, or a key-derivation function. Many AEAD designs use a MAC as an internal component and rely on its security to provide integrity guarantees. Recently, Menda et al. [MLGR23] showed that for some AEADs following a particular format to be committing, they require their underlying MAC to be (constrained) preimage resistant as it would otherwise lead to a committing attack. Similarly, Farshim et al. [FOR17] and Grubbs et al. [GLR17] analyzed probabilistic generic AEAD compositions. Farshim et al. showed that combining a key-committing secure MAC with a key-committing secure symmetric encryption scheme is sufficient for the composition to be key-committing secure. Grubbs et al. analyzed a different security notion called receiver binding.

Due to this intrinsic relationship between the committing security of an AEAD and its underlying MAC, it is natural to wonder whether one could strengthen an AEAD to be context-committing by simply strengthening its underlying MAC without adding an extra primitive. The generic composition analyses by [FOR17, GLR17] are a first step in this direction but are limited in some aspects. Firstly, they focus on the weaker notions of key-committing and receiver binding, and nothing is known about generic composition for context-committing or context-discovery secure AEAD. Secondly, the analyses are confined to probabilistic AE and do not consider the modern formalism of nonce-based AE with associated data. As shown in [NRS14], in this case, more paradigms than the classical three Encrypt-then-MAC, MAC-then-Encrypt and Encrypt-and-MAC are possible and secure. Thirdly, it requires some strong and sometimes unrealistic assumptions on the components. [FOR17] requires that the symmetric encryption composed is key-committing secure, which, as we shall demonstrate, is never the case for IV-based symmetric encryption schemes. [GLR17] requires the MAC to be a random oracle in the analysis of MAC-then-Encrypt and does not analyze Encrypt-and-MAC.

As outlined, previous studies on committing security considered the committing security of MACs solely in the context of AEAD. In hindsight, this approach seems shortsighted, given the widespread utilization of MACs as standalone primitives across diverse contexts, rendering them susceptible to misuse akin to AEAD. Furthermore, even within the AEAD context, if one were to look for a secure instantiation of a committing MAC, the general status of real-world MAC schemes remains elusive. Although a plethora of standardized MAC schemes exists, they were never analyzed through the point of view of commitment. In this work, we remedy this state of affairs by providing a comprehensive analysis of MAC committing security. We further motivate this need by providing concrete settings beyond AEAD where committing and context-discovery secure MACs are required. As an application to AEAD, we present, via nonce-based generic compositions, simple and efficient ways to build committing and context-discovery secure AEAD schemes. As an added benefit, these solve the previously identified issue of reducing the number of primitives employed while facilitating the reuse of legacy MAC and symmetric encryption implementations. Below is a summary of our contributions.

1.1 Our Contributions

Settings Requiring MAC Commitment and CDY Security. We start by motivating the need for committing secure MACs by identifying a few real-world scenarios that require it to provide the expected security guarantee. In parallel to commitment, we found the first real-world setting requiring a restricted form of context-discovery security. As commitment security does not imply context-discovery security (similar to collision resistance not implying preimage resistance), this motivates the need for more robust security for MAC schemes than only context-commitment, especially the need for context-discovery secure MACs.

Committing and CDY Security Notions for MACs. We formalize the required MAC security properties previously identified: committing and context-discovery security. In addition, we provide various restricted and strengthened versions of these notions and how they relate to each other. It allows one to better understand the hierarchy between the different notions and more easily identify which notion provides the combination of required security guarantees for a target application. This leads us to define a stronger security notion that implies all the others and for which we believe MAC designers should aim at. We call this notion *strong-committing* security (SCMT) and define it as the combination of context-committing and context-discovery security. We further show how the notions of committing AEADs relate to our notions of committing MACs.

Scheme	CMT_k	CMT	CDY	Analysis
CBC-type MACs [ISO11a]	no	no	no	Sec. 5.1
HMAC with variable-length keys [ISO21]	no	no	?	Sec. 5.2
Badger [ISO11b]	no	no	no	Sec. 5.3
Poly1305-AES [ISO11b]	no	no	no	Sec. 5.3
GMAC [ISO11b]	no	no	no	Sec. 5.3
LightMAC [ISO19]	no	no	no	Sec. 5.4
Chaskey [ISO19]	no	no	no	Sec. 5.4
CBC-MAC-C1 [this work]	yes	no	yes	Thm 1
CMAC-C1 [this work]	yes	no	yes	Thm 3
HMAC with fixed-length keys [ISO21]	yes	yes	yes	Thm 5

Figure 1: Summary of MAC schemes security. CMT_k, CMT and CDY means respectively key-committing, context-committing and context-discovery secure.

Committing and CDY Security Analysis of MAC Constructions. Using our newly defined MAC security notions, we analyze the security of various popular and standardized MAC schemes. The primary outcome of this analysis is that except for HMAC with fixed-length key that is context-committing secure, most other MACs analyzed are vulnerable to key-committing or context-discovery attacks. We provide efficient fixes to these schemes to guarantee key-commitment security when possible. We summarize these results in Fig. 1.

Application to Generic Composition for Nonce-Based AEAD. While we showed above that committing and context-discovery secure MACs are of practical interest on their own, a natural question is how this security transfers to AEAD schemes built from the generic composition of a MAC and an encryption scheme. While this was studied for probabilistic AEAD schemes [FOR17, GLR17], with the limitations described in the introduction, nothing is known about the nonce-based case in general. We fill this gap in the literature on committing AEAD schemes by analyzing the nonce-based generic AEAD compositions from [NRS14]. For most cases, we identify sufficient requirements on the MAC and the encryption scheme for the AEAD composition to be committing and/or context-discovery secure. We also analyze these compositions when combined with a key-schedule function, which is used to derive the MAC and encryption keys.

Combined with the above result on the committing security of practical MAC constructions, our generic composition analyses help to understand better why AEAD schemes following these paradigms—sometimes loosely—are vulnerable to committing attacks. Some examples of such schemes are the standards in ISO/IEC 19772 [ISO20] and in IEEE Std 1619.1 [IEE19] that strictly follow Encrypt-then-MAC, the standards in RFC 5297 [Har08] and ISO 20038 [ISO17] that strictly follows the SIV paradigm. Standards that loosely follow these paradigms are GCM [Dwo07a], ChaCha20-Poly1305 [NL18] and EAX [BRW04] that loosely follow Encrypt-then-MAC, CCM [Dwo07b] that loosely follows MAC-then-Encrypt, and OCB [KR14] that loosely follows Encrypt-and-MAC. Our analyses also help to understand better which fixes are needed on the MAC or the encryption part of these schemes to make them committing secure.

Our generic composition analyses are also profitable in providing simple and generic ways to obtain context-committing secure AEADs (with a proper key schedule) that are only one pass over the associated data without an extra primitive. In contrast, all of the suggested solutions that are not duplex-based need two passes or an extra primitive [BH22, CR22].

2 Preliminaries

Notation. The empty string is denoted by ε. For any positive integer n, $\{0,1\}^n$, $\{0,1\}^{\leq n}$ and $\{0,1\}^*$ denote respectively the set of binary strings of size n, the set of binary strings of size less than or equal to n, and the set of all finite strings. For any string x, $|x|$ denotes its size in bits and $|x|_n = \lceil |x|/n \rceil$ its size in n-bit blocks. For $1 \leq i < j \leq |x|$, let $x[i{:}j]$ denote the substring of x spanning bit i to bit j inclusive, and $x \gg n$ denote the substring $x[1{:}|x| - n]$. For a positive integer $n \leq |u|$, we use $\lfloor u \rfloor_n$ to denote the strings obtained by truncating u to its leftmost n bits. For any two strings x and y, $x \| y$ and $x \oplus y$ denote their concatenation and bitwise XOR respectively. For any positive integer c such that $|x| = |y| = c \cdot n$, $x +_n y$ and $x -_n y$ denote respectively the strings obtained from interpreting their n-bit substrings as unsigned integers and adding or subtracting individually modulo 2^n. For $n \in \mathbb{N}$ and $u \in \{0,1\}^*$, $(u_1, \ldots, u_\ell) \overset{n}{\leftarrow} u$ denotes the n-bit parsing of u where $|u_i| = n$ for all $1 \leq i < \ell$ and $0 < |u_i| \leq n$. For $a, b \in \mathbb{N}$, if $a < 2^b$, then $\langle a \rangle_b$ denotes the b-bit binary representation of a.

Unless otherwise stated, an algorithm may be randomized. For any algorithm A we use $y \leftarrow \mathsf{A}(x_1, x_2, \ldots)$ to denote the process of running A on the indicated inputs and fresh random coins, and assigning the output to y. By convention, the running time of an adversary refers to the sum of its actual running time and the size of its description. We generically refer to the resources of an adversary as any subset of the following quantities: its running time, the number of queries that it makes to its oracles, and the total length (in bits) of its oracle queries. We write $\mathsf{B.sub1}$, $\mathsf{B.sub2}, \ldots$ to denote a group of algorithms that share states and refer to them collectively as B.

Message Authentication Code. A message authentication code $\mathsf{MAC} = (\mathsf{MAC.Tg}, \mathsf{MAC.Vf})$ is a pair of algorithms such that:

- The possibly randomized tagging algorithm $\mathsf{MAC.Tg} : \mathcal{K}_m \times \mathcal{M} \to \mathcal{T}$ takes as input a secret key $K \in \mathcal{K}_m$, a message $M \in \mathcal{M}$ and returns a tag $T \in \mathcal{T}$.

- The deterministic verification algorithm $\mathsf{MAC.Vf} : \mathcal{K}_m \times \mathcal{M} \times \mathcal{T} \to \{0,1\}$ takes as input a secret key $K \in \mathcal{K}_m$, a message $M \in \mathcal{M}$, and a tag $T \in \mathcal{T}$ and returns a bit $b \in \{0,1\}$, where $b = 0$ indicates an invalid tag.

We refer to the associated sets $\mathcal{K}_m$, $\mathcal{M}$, and $\mathcal{T}$ as the MAC key space, the message space, and the tag space, respectively. We require every MAC to satisfy correctness, namely for all $(K, M) \in (\mathcal{K}_m, \mathcal{M})$, it must hold that if $\mathsf{MAC.Tg}(K, M) = T$ then $\mathsf{MAC.Vf}(K, M, T) = 1$.

Nonce-Based MAC. Similarly to a MAC, a nonce-based message authentication code $\mathsf{MAC} = (\mathsf{MAC.Tg}, \mathsf{MAC.Vf})$ is a pair of deterministic algorithms that takes an additional input $N \in \mathcal{N}$ called a nonce, i.e., $\mathsf{MAC.Tg} : \mathcal{K}_m \times \mathcal{N} \times \mathcal{M} \to \mathcal{T}$ and $\mathsf{MAC.Vf} : \mathcal{K}_m \times \mathcal{N} \times \mathcal{M} \times \mathcal{T} \to \{0,1\}$. Again, we require the correctness property: for all $(K, N, M) \in (\mathcal{K}_m, \mathcal{N}, \mathcal{M})$, it must hold that if $\mathsf{MAC.Tg}(K, N, M) = T$ then $\mathsf{MAC.Vf}(K, N, M, T) = 1$.

Canonical Verification. We say that a nonce-based MAC is with the canonical verification algorithm when $\mathsf{MAC.Vf}$ satisfies the following definition: for all $(K, N, M, T) \in (\mathcal{K}_m, \mathcal{N}, \mathcal{M}, \mathcal{T})$, $\mathsf{MAC.Vf}(K, N, M, T)$ returns 1 if $\mathsf{MAC.Tg}(K, N, M) = T$, and 0 otherwise. A similar definition applies to deterministic MACs without nonces by simply removing the nonces from the definition.

IV-Based Symmetric-key Encryption (ivE). An IV-based symmetric-key encryption scheme $\mathsf{SE} = (\mathsf{SE.Enc}, \mathsf{SE.Dec})$ is a pair of algorithms such that:

- The deterministic encryption algorithm $\mathsf{SE.Enc} : \mathcal{K}_e \times \mathcal{IV} \times \mathcal{M} \to \mathcal{C}$ takes as input a secret key $K \in \mathcal{K}_e$, an initial vector $IV \in \mathcal{IV}$, and a message $M \in \mathcal{M}$ and returns a ciphertext $C \in \mathcal{C}$.

- The deterministic decryption algorithm $\mathsf{SE.Dec} : \mathcal{K}_e \times \mathcal{IV} \times \mathcal{C} \to \mathcal{M}$ takes as input a secret key $K \in \mathcal{K}_e$, an initial vector $IV \in \mathcal{IV}$, associated data $A \in \mathcal{AD}$, and a ciphertext $C \in \mathcal{C}$ and returns a message $M \in \mathcal{M}$.

We require every IV-based symmetric-key encryption to satisfy correctness and tidiness, namely for all $(K, IV, M) \in (\mathcal{K}_e, \mathcal{IV}, \mathcal{M})$, it must hold that if $\mathsf{SE.Enc}(K, IV, M) = C$ then $\mathsf{SE.Dec}(K, IV, C) = M$ and vice versa. Furthermore, when $\mathsf{SE.Enc}(K, IV, M) = C$, we assume $|M| = |C|$ in this paper. Note that in this case, $\mathcal{M}$ is isomorphic to $\mathcal{C}$.

To guarantee security, an IV-based symmetric-key encryption scheme might require either that the initial vector IV be random for each encryption query, or to not repeat across different encryption queries. In this last case, it is also called nonce-based symmetric-key encryption.

Nonce-Based AEAD. A nonce-based authenticated encryption scheme with associated data $\mathsf{AE} = (\mathsf{AE.Enc}, \mathsf{AE.Dec})$ is a pair of algorithms such that:

- The deterministic encryption algorithm $\mathsf{AE.Enc} : \mathcal{K} \times \mathcal{N} \times \mathcal{AD} \times \mathcal{M} \to \mathcal{C}$ takes as input a secret key $K \in \mathcal{K}$, a nonce $N \in \mathcal{N}$, associated data $A \in \mathcal{AD}$, and a message $M \in \mathcal{M}$ and returns a ciphertext $C \in \mathcal{C}$.
- The deterministic decryption algorithm $\mathsf{AE.Dec} : \mathcal{K} \times \mathcal{N} \times \mathcal{AD} \times \mathcal{C} \to \mathcal{M} \cup \{\perp\}$ takes as input a secret key $K \in \mathcal{K}$, a nonce $N \in \mathcal{N}$, associated data $A \in \mathcal{AD}$, and a ciphertext $C \in \mathcal{C}$ and returns either a message $M \in \mathcal{M}$ or the symbol $\perp$ to indicate an invalid ciphertext.

We refer to the associated sets $\mathcal{K}$, $\mathcal{N}$, $\mathcal{AD}$, $\mathcal{M}$, and $\mathcal{C}$ as the key space, the nonce space, the associated-data space, the message or plaintext space, and the ciphertext space, respectively. We require every nonce-based AEAD to satisfy correctness, namely for all $(K, N, A, M) \in (\mathcal{K}, \mathcal{N}, \mathcal{AD}, \mathcal{M})$, it must hold that if $\mathsf{AE.Enc}(K, N, A, M) = C$ then $\mathsf{AE.Dec}(K, N, A, C) = M$. We say that AE is tidy, if for all $(K, N, A, C) \in (\mathcal{K}, \mathcal{N}, \mathcal{AD}, \mathcal{C})$ it holds that if $\mathsf{AE.Dec}(K, N, A, C) = M \neq \perp$ then $\mathsf{AE.Enc}(K, N, A, M) = C$.

PRF Security. Let $F : \mathcal{K} \times \{0,1\}^* \to \{0,1\}^n$ be a keyed function. Let $\mathcal{R} : \{0,1\}^* \to \{0,1\}^n$ be a random function. The advantage of $\mathcal{A}$ against the PRF security of F is defined as

$$\mathsf{Adv}_F^{\mathrm{prf}}(\mathcal{A}) = \left|\Pr[\mathcal{A}^{F_K} = 1] - \Pr[\mathcal{A}^{\mathcal{R}} = 1]\right|$$

where K is chosen uniformly at random from $\mathcal{K}$.

Collision Resistance and Preimage Resistance. We consider collision resistance and everywhere preimage resistance [RS04] for a hash function $H : \mathcal{M} \to \mathcal{Y}$ as follows.

Definition 1. *Let $H : \mathcal{M} \to \mathcal{Y}$ be a hash function and let $\mathcal{A}$ be an adversary. Then we define*

$$\mathsf{Adv}_H^{\mathrm{coll}}(\mathcal{A}) = \Pr[(M, M') \leftarrow\!\!\$\, \mathcal{A} : (M \neq M') \wedge (H(M) = H(M'))].$$

Definition 2. *Let $H : \mathcal{M} \to \mathcal{Y}$ be a hash function and let $\mathcal{A}$ be an adversary. Then we define*

$$\mathsf{Adv}_H^{\mathrm{epre}}(\mathcal{A}) = \max_{Y \in \mathcal{Y}} \{\Pr[M \leftarrow\!\!\$\, \mathcal{A} : H(M) = Y]\}.$$

3 Practical Applications of Committing and CDY MACs

We point out four practical settings where MACs are used. For one of them, the MAC is explicitly required to satisfy random-key robustness, which is a weak version of key commitment. Conversely, while not explicitly stated, the expectation for the underlying MACs in the three remaining settings is that they be committing or context-discovery secure. This tacit assumption underscores the inherent expectation for MACs to satisfy these properties.

Authentication without key identification. In authentication protocols using a MAC as an underlying primitive, such as in ISO/IEC 9798-4 [ISO99], it is usually not required in the specification that the party sending the MAC digest identify which key has been used for the computation of the MAC, assuming that the receiver can unambiguously identify which key to use for verification. This problem has already been noticed for authenticated encryption by Len et al. [LGR22]. We provide below a concrete example of where this could be exploited.

Some RFID tag authentication protocols, such as protocols standardized in ISO/IEC 29167-11 [ISO23], use a challenge-response protocol. To authenticate a tag, an RFID reader generates a nonce N and sends it to the tag. The tag then computes $T = \mathsf{MAC.Tg}(K, N)$ using a shared secret key K and returns T to the reader. The reader, who also knows the secret key K, computes $T' = \mathsf{MAC.Tg}(K, N)$ and checks if $T = T'$ to authenticate the tag. A problem that arises if the reader has multiple secret keys, K_1, K_2, $\ldots$, is determining which key to use to verify the MAC. For ISO/IEC 29167-11, this is "a matter of key management that lies outside the scope" of the document. A proposed solution to solve this problem is key searching [WSRE04, MW04, MSW06, TSL08, FZOeS14] which consists of testing all secret keys until we find the "right" key that verifies the tag. The problem of these approaches is that they rely on the assumption that the MAC used is key-committing. Adapted from the AEAD notions, key-committing security for a PRF-based MAC can be simply defined by the difficulty to find a pair $(K_1, M_1), (K_2, M_2)$ satisfying $K_1 \neq K_2$ such that $\mathsf{MAC.Tg}(K_1, M_1) = \mathsf{MAC.Tg}(K_2, M_2)$.

A possible attack scenario in the case of a supply chain management, where authentication of packages is used for traceability, is to have a packet that returns a malicious tag satisfying $T = \mathsf{MAC.Tg}(K_1, N) = \mathsf{MAC.Tg}(K_2, N)$ for two keys. When the package is scanned at a warehouse, two checks are done through two different suppliers with which the tag shares one of the keys. The first supplier attests to the packet's origin with key K_1, whereas the second supplier provides the new destination for the package with key K_2. For each supplier, the "correct" key for the packet is found through key searching. An unexpected vulnerability of such a supply chain process is that a malicious party could illegitimately claim the loss of the packet identified with key K_1. Retracing the path of this packet through suppliers would then show that the packet arrived at the warehouse but that there is no trace of it leaving the warehouse, invalidating, therefore, the purpose of the supply chain management.

The OPAQUE Asymmetric PAKE Protocol [BKLW23]. The OPAQUE protocol is one of the password-authenticated key exchanges (PAKE) recommended by the CFRG for usage in IETF protocols. In its originally proposed cryptographic design [JKX18], OPAQUE required its underlying authenticated encryption scheme to satisfy the *random-key robustness* property, which is a property implied by key-commitment. In the latest version of the IRTF draft specifying OPAQUE [BKLW23], this requirement has been replaced by a similar requirement, but now only on the underlying MAC. For a MAC to be random-key robust, given two randomly generated keys K_1 and K_2, it should be hard for an adversary to find a message M such that $\mathsf{MAC.Tg}(K_1, M) = \mathsf{MAC.Tg}(K_2, M)$. HMAC is proposed as an example of such a secure MAC scheme, while GMAC is considered insecure. As we will see in the next section, this key robustness property for MAC is naturally implied by key-committing security, highlighting the relevance of key-committing MACs. Moreover, the proposed instantiation by HMAC can be seen as costly and non-flexible, as no alternative is offered. Our analyses of dedicated MAC schemes in Sec. 5, give alternative instantiation based on different primitives than a hash function.

Collision Resistant KDF. KDFs are generally used to generate uniform random keys from nonperfect randomness or to derive multiple sub-keys from a master key. In both cases, in addition to their primary goal, they are often required to be collision-resistant. However, to provide this property when they are based on a MAC or a PRF, they require some form of committing security from their underlying primitive.

If we consider HKDF [KE10], which is a KDF based on HMAC, it was shown in [Stä22] that the lack of key-commitment in HMAC makes it possible to find collisions in HKDF, which can be exploited to attack the age command line encryption.

In the case of sub-keys derivation, a PRF/MAC can be used to derive a sub-key $K_{\text{OUT}} = \mathsf{MAC.Tg}(K_{\text{IN}}, \text{Ctx})$, with as inputs the master key K_{IN} and a context Ctx. The context Ctx enables the derivation of multiple sub-keys for the same master key K_{IN}. It usually contains data that identifies the purpose of the derived sub-key and other related information, such as the identity of the different parties using the sub-key or a nonce to prevent reuse. HMAC, CMAC, and KMAC are the three MACs recommended for this purpose in NIST SP 800-108r1 [Che22]. In the same document, it is recommended to do *context binding* when deriving sub-keys, which consists of including the information just described above in Ctx. The goal of context binding is to provide "assurance that all parties who (correctly) derive the keying material share the same understanding of who will access it and in which session it will be used. If those parties have different understandings, then they will derive different keying material." It is therefore expected that it is hard to obtain $(K_{\text{IN}}, \text{Ctx}) \neq (K'_{\text{IN}}, \text{Ctx}')$ such that $\mathsf{MAC.Tg}(K_{\text{IN}}, \text{Ctx}) = \mathsf{MAC.Tg}(K'_{\text{IN}}, \text{Ctx}')$. This property is precisely our new notion of context-committing security for MACs and is usually not guaranteed by the conventional MAC security notion. As we will show in Sec. 5, CMAC—one of the recommended schemes—is not context-committing.

To be noted, collision-resistant PRFs and KDFs are often also recommended as a solution to fix non-committing AEAD schemes [FOR17, GLR17, ADG+22].

Timed Efficient Stream Loss-Tolerant Authentication (TESLA) [PCS+05]. While the previous settings highlighted the need for key-committing MACs, we now exhibit a real-world protocol that requires a restricted form of context-discovery security (defined as key-discovery security later in Sec. 4.1). TESLA is a scheme used to provide sender authentication for multicast and broadcast streams. It was proposed by Perrig et al. [PCTS00] and is defined in RFC 4082 [PCS+05]. During this protocol, a sender wants to broadcast a sequence of messages $M_1, \cdots, M_\ell$ to multiple receivers and guarantee these messages' authentication.

At its core, TESLA uses a one-way chain where the sender randomly chooses the last element of the chain $K_{\ell+1}$ and generates a key chain $K_0, K_1, \cdots, K_{\ell+1}$ by applying repeatedly a function F (i.e., $K_i = F(K_{i+1})$, cf. Fig. 2). The key K_0 is then shared with all the receivers through an authenticated channel during the initialization phase of the protocol. Once the initialization is done, for each key K_i, an authentication key $K'_i = F'(K_i)$ is derived through a one way function F' and used to compute $\mathsf{MAC.Tg}(K'_i, M_i)$. Then the message packets are sent sequentially in the following order—first $(M_1, \mathsf{MAC.Tg}(K'_1, M_1), K_0)$, then $(M_2, \mathsf{MAC.Tg}(K'_2, M_2), K_1)$, etc., until $(M_\ell, \mathsf{MAC.Tg}(K'_\ell, M_\ell), K_{\ell-1})$. Upon reception of a packet (M_i, T_i, K_{i-1}), a receiver waits for the next packet (M_{i+1}, T_{i+1}, K_i) to be able to compute $K'_i = F'(K_i)$ and verifies that $\mathsf{MAC.Vf}(K'_i, M_i, T_i) = 1$ and $K_0 = F^i(K_i)$. This process is called a delayed message authentication protocol.

As K_0 has been broadcasted with the first packet, essential for the security of the protocol, is the one-wayness property of F as otherwise, an adversary can come up with a forged key K_i^{forge} that satisfies $K_0 = F^i(K_i^{\text{forge}})$. For the security of the protocol, the K_i must also be random strings (cf. [PCTS00]). Thus, TESLA instantiates the function F with a PRF f_k (HMAC in their implementation). This function is then defined as $F(x) = f_x(0)$, and a sufficient condition on f_k according to [PCTS00] is that for a randomly chosen K, an adversary which is given $f_K(0)$ is unable to find $K' \neq K$ such that $f_{K'}(0) = f_K(0)$. This property would be naturally guaranteed if f_k was key-committing, but a closer look shows that the condition $K' \neq K$ does not exclude valid adversaries and attacks that can find $f_{K'}(0) = f_K(0)$ with $K' = K$. In the following section, we formalize this security requirement under the name of key-discovery security and note that the committing notions do not generally imply this notion.

$$\begin{tikzcd} K_0 & K_1 \arrow[l, "F(K_1)"'] & \cdots \arrow[l, "F(K_2)"'] & \cdots & K_\ell \arrow[l, "F(K_\ell)"'] & K_{\ell+1} \arrow[l, "F(K_{\ell+1})"'] \arrow[lllll, dashed, bend right, "\textbf{Generate}"'] \\ {} \arrow[rrrrr, dashed, "\textbf{Reveal}"] & & & & & {} \end{tikzcd}$$

Figure 2: One-way key chain example.

$$\begin{tikzcd} \mathrm{RBT}_k & \mathrm{CMT}_k \arrow[l] & \mathrm{CMT} \arrow[l] \arrow[dl, dashed] & \mathrm{SCMT} \arrow[l] \arrow[dl] \\ & \mathrm{CDY}_k & \mathrm{CDY} \arrow[l, dashed] & \end{tikzcd}$$

Figure 3: Relations between commitment and context-discovery notions for MACs and AEADs. A solid arrow $A \rightarrow B$ means that any scheme that is A-secure is also B-secure, dotted arrows hold only for a restricted class of context-selector.

4 Context-Discovery and Committing Security Notions

In this section, we provide the notions of context-committing (CMT) and context-discovery security (CDY) for MACs, laying the groundwork for a deeper understanding of their cryptographic properties. Additionally, we introduce weaker variants that are easier to achieve, namely key-committing (CMT_k) and key-discovery security (CDY_k), and a stronger committing notion (SCMT). The relations between the notions are illustrated in Fig. 3.

In anticipation of the forthcoming generic composition analyses, we also recall the corresponding security notions for AEADs that reformulate the notions from [BH22, MLGR23]. Furthermore, recognizing that IV-based symmetric-key encryption schemes do not satisfy key-commitment, we propose a weaker notion termed key-robustness (RBT_k) and show that classical symmetric encryption modes satisfy this property.

4.1 Security Notions for MACs

We present a comprehensive set of commitment and context-discovery security notions for MACs. To the best of our knowledge, the context-discovery notions for MACs are new, while some of the committing notions are already present in the literature, albeit under different names.

In the following security games, $\mathsf{MAC} = (\mathsf{MAC.Tg}, \mathsf{MAC.Vf})$ is a nonce-based message authentication code. Further, we define S to be a randomized algorithm, called context-selector, that takes no input and outputs a challenge value given to the adversary at the beginning of the game. It is used to model a variety of scenarios where an adversary is able to obtain a challenge from a protocol that is computed exactly as the context-selector.

Context-Discovery Notions for MACs. The following notions are adaptations of the AEAD context-discovery notions from [MLGR23].

Context-discovery game CDY **for MACs.** In the MAC-context-discovery game, a challenge tag $T \in \mathcal{T}$ is computed by the context-selector S and given to the adversary $\mathcal{A}$. This adversary outputs (K, N, M), and wins if $\mathsf{MAC.Tg}(K, N, M) = T$. We write $\mathsf{Adv}^{\mathrm{macCDY}}_{\mathsf{MAC}}(\mathcal{A}; \mathsf{S})$ to denote the probability that $\mathcal{A}$ wins for the context-selector S.

The context-discovery game for a MAC is equivalent to the notions of preimage resistance for a hash function [RS04]. The context-selector is a parameter to model different scenarios, i.e., how the challenge is computed. Parametrizing the game with a context-selector S allows us to model both the classical notions of range-oriented and everywhere preimage resistance as well as the other variants of preimage resistance [AS11]. We require $\mathsf{Adv}^{\mathrm{macCDY}}_{\mathsf{MAC}}(\mathcal{A};\mathsf{S})$ to be small only for a restricted set of context-selectors that would model the threat scenario.

Key-discovery game CDY_k **for MACs.** In the MAC-key-discovery game, a challenge tuple $(N, M, T) \in \mathcal{N} \times \mathcal{M} \times \mathcal{T}$ is computed by the context-selector S and given to the adversary $\mathcal{A}$. This adversary outputs K, and wins if $\mathsf{MAC.Tg}(K, N, M) = T$. We write $\mathsf{Adv}^{\mathrm{macCDY_k}}_{\mathsf{MAC}}(\mathcal{A};\mathsf{S})$ to denote the probability that $\mathcal{A}$ wins for the context-selector S.

This game can notably model the security required by the MAC used in the TESLA protocol analyzed in the previous section. The context-selector that would model the required security for TESLA, would sample a random key K and return a challenge tuple $(0, T)$ where T is computed as $T = \mathsf{MAC.Tg}(K, 0)$.

Committing Notions for MACs. The following context-committing and key-committing notions are adaptations of the CMT-4 and CMT-1 notions from [BH22]. The two remaining notions are new.

Context-committing game CMT **for MACs.** In the MAC-context-committing game, an adversary $\mathcal{A}$ outputs (K_1, N_1, M_1) and (K_2, N_2, M_2); $\mathcal{A}$ wins if:

- $(K_1, N_1, M_1) \neq (K_2, N_2, M_2)$;
- $\mathsf{MAC.Tg}(K_1, N_1, M_1) = \mathsf{MAC.Tg}(K_2, N_2, M_2)$.

We write $\mathsf{Adv}^{\mathrm{macCMT}}_{\mathsf{MAC}}(\mathcal{A})$ to denote the probability that $\mathcal{A}$ wins.

In the literature, this context-committing notion is sometimes called collision-resistance, for example, in [GLR17].

Key-committing game CMT_k **for MACs.** In the MAC-key-committing game, an adversary $\mathcal{A}$ outputs (K_1, N_1, M_1) and (K_2, N_2, M_2); $\mathcal{A}$ wins if:

- $K_1 \neq K_2$;
- $\mathsf{MAC.Tg}(K_1, N_1, M_1) = \mathsf{MAC.Tg}(K_2, N_2, M_2)$.

We write $\mathsf{Adv}^{\mathrm{macCMT_k}}_{\mathsf{MAC}}(\mathcal{A})$ to denote the probability that $\mathcal{A}$ wins.

A similar notion is defined in [FOR17] under the name of full robustness (FROB).

Key-robustness game RBT_k **for MACs.** In the MAC-key-robustness game, an adversary $\mathcal{A}$ outputs (K_1, K_2, N, M); $\mathcal{A}$ wins if:

- $K_1 \neq K_2$;
- $\mathsf{MAC.Tg}(K_1, N, M) = \mathsf{MAC.Tg}(K_2, N, M)$.

We write $\mathsf{Adv}^{\mathrm{macRBT_k}}_{\mathsf{MAC}}(\mathcal{A})$ to denote the probability that $\mathcal{A}$ wins.

This weaker notion than key-committing models, for example, attack scenarios where the data storage and the key management system (KMS) are separate. An adversary may be able to exploit the KMS by adding a key K_2 distinct from the genuine key K_1, but not to temper with the data and its tag. If the MAC used is not key-robust secure then this malicious key K_2 could also be used to authenticate the data. Note that this notion is slightly stronger than the random-key robustness notion required by the underlying MAC in OPAQUE [BKLW23].

Strong-committing game SCMT **for MACs.** In the MAC-strong-committing game, a challenge tag $T \in \mathcal{T}$ is computed by the context-selector S and given to the adversary $\mathcal{A}$. This adversary outputs $(K_1, N_1, M_1), (K_2, N_2, M_2)$, and wins if either $\mathsf{MAC.Tg}(K_1, N_1, M_1) = T$, or the following hold:

- $(K_1, N_1, M_1) \neq (K_2, N_2, M_2)$;
- $\mathsf{MAC.Tg}(K_1, N_1, M_1) = \mathsf{MAC.Tg}(K_2, N_2, M_2)$.

We write $\mathsf{Adv}^{\mathrm{macSCMT}}_{\mathsf{MAC}}(\mathcal{A}; \mathsf{S})$ to denote the probability that $\mathcal{A}$ wins for the context-selector S.

The SCMT notion is equivalent to combining the CDY and CMT notions. It implies all the MAC security notions presented in this section and has a compact description. As we will show in the generic composition section, it facilitates proof that shows simultaneously CDY and CMT security.

Relations. We provide in Fig. 3 the relations between the different notions. The justification of these relations is given in App. A.

Related Notions. Similar security definitions as above apply to MACs without nonces by simply removing the nonces from the definitions. These definitions then also apply to pseudorandom functions (PRFs) as they have the same syntax as a nonce-less MAC tagging function $\mathsf{MAC.Tg}$.

Moreover, similarly to the difference between the CMT and the CMTD notions from [BH22] that defines commitment either through the encryption or the decryption function, all the previously presented MAC security notions can be defined through the verification function $\mathsf{MAC.Vf}$ instead of the tagging function $\mathsf{MAC.Tg}$. We denote these V-notions: $\mathrm{CDYV}, \mathrm{CDYV}_k, \mathrm{CMTV}, \mathrm{CMTV}_k, \mathrm{RBTV}_k$ and SCMTV. In a similar way as for AEADs, the V-notions imply their corresponding normal version due to the correctness property of MAC. Conversely, the normal notions imply their corresponding V-version only if MAC is with the canonical verification algorithm.

4.2 Security Notions for AEADs

We now give corresponding AEAD notions to the MAC notions given in the previous section.

In the following security games, $\mathsf{AE} = (\mathsf{AE.Enc}, \mathsf{AE.Dec})$ is a nonce-based authenticated encryption scheme with associated data. Similarly as for MACs, we define the context-selector S as a randomized algorithm that takes no input, and outputs a challenge value given to the adversary at the beginning of the game.

Context-Discovery Notions for AEADs. The two following notions are simplified versions of the AEAD context-discovery notions from [MLGR23].

Context-discovery game CDY **for AEADs.** In the AEAD-context-discovery game, a challenge ciphertext $C \in \mathcal{C}$ is computed by the context-selector S and given to the adversary $\mathcal{A}$. This adversary outputs (K, N, A, M), and wins if $\mathsf{AE.Enc}(K, N, A, M) = C$. We write $\mathsf{Adv}^{\mathrm{aeadCDY}}_{\mathsf{AE}}(\mathcal{A}; \mathsf{S})$ to denote the probability that $\mathcal{A}$ wins for the context-selector S.

Key-discovery game CDY_k **for AEADs.** In the AEAD-key-discovery game, a challenge tuple $(N, A, M, C) \in \mathcal{N} \times \mathcal{AD} \times \mathcal{M} \times \mathcal{C}$ is computed by the context-selector S and given to the adversary $\mathcal{A}$. This adversary outputs K, and wins if $\mathsf{AE.Enc}(K, N, A, M) = C$. We write $\mathsf{Adv}^{\mathrm{aeadCDY_k}}_{\mathsf{AE}}(\mathcal{A}; \mathsf{S})$ to denote the probability that $\mathcal{A}$ wins for the context-selector S.

Committing Notions for AEADs. The two following context-committing and key-committing notions are equivalent to the CMT-3 and CMT-1 notions from [BH22].

Context-committing game CMT **for AEADs.** In the AEAD-context-committing game, an adversary $\mathcal{A}$ outputs (K_1, N_1, A_1, M_1) and (K_2, N_2, A_2, M_2); $\mathcal{A}$ wins if:

- $(K_1, N_1, A_1) \neq (K_2, N_2, A_2)$;
- $\mathsf{AE.Enc}(K_1, N_1, A_1, M_1) = \mathsf{AE.Enc}(K_2, N_2, A_2, M_2)$.

We write $\mathsf{Adv}_{\mathsf{AE}}^{\mathrm{aeadCMT}}(\mathcal{A})$ to denote the probability that $\mathcal{A}$ wins.

As proved in [BH22], CMT-3 ($(K_1, N_1, A_1) \neq (K_2, N_2, A_2)$) is equivalent to CMT-4 ($(K_1, N_1, A_1, M_1) \neq (K_2, N_2, A_2, M_2)$) for AEAD. This is because the message M is implicitly contained in the ciphertext when (K, N, A) is fixed. Note that there is no similar relation for MACs. In this paper, we consider the CMT-3 notion as it is the simpler notion.

Key-committing game CMT_k **for AEADs.** In the AEAD-key-committing game, an adversary $\mathcal{A}$ outputs (K_1, N_1, A_1, M_1) and (K_2, N_2, A_2, M_2); $\mathcal{A}$ wins if:

- $K_1 \neq K_2$;
- $\mathsf{AE.Enc}(K_1, N_1, A_1, M_1) = \mathsf{AE.Enc}(K_2, N_2, A_2, M_2)$.

We write $\mathsf{Adv}_{\mathsf{AE}}^{\mathrm{aeadCMT_k}}(\mathcal{A})$ to denote the probability that $\mathcal{A}$ wins.

The following key-robustness (RBT_k) and strong-committing (SCMT) notions are two new notions defined in this paper. The first one is weaker than key-committing and will be useful for the analysis of generic compositions in Sec. 6. The second one is a simple combination of the context-committing and context-discovery security notion for AEAD. In case of standardization of a committing AEAD, we believe that the aimed security notion should be SCMT as a future-proof precaution.

Key-robustness game RBT_k **for AEADs.** In the AEAD-key-robustness game, an adversary $\mathcal{A}$ outputs (K_1, K_2, N, A, M); $\mathcal{A}$ wins if:

- $K_1 \neq K_2$;
- $\mathsf{AE.Enc}(K_1, N, A, M) = \mathsf{AE.Enc}(K_2, N, A, M)$.

We write $\mathsf{Adv}_{\mathsf{AE}}^{\mathrm{aeadRBT_k}}(\mathcal{A})$ to denote the probability that $\mathcal{A}$ wins.

Strong-committing game SCMT **for AEADs.** In the AEAD-strong-committing game, a challenge ciphertext $C \in \mathcal{C}$ is computed by the context-selector S and given to the adversary $\mathcal{A}$. This adversary outputs $(K_1, N_1, A_1, M_1), (K_2, N_2, A_2, M_2)$ and wins if either $\mathsf{AE.Enc}(K_1, N_1, A_1, M_1) = C$, or the following hold:

- $(K_1, N_1, A_1) \neq (K_2, N_2, A_2)$;
- $\mathsf{AE.Enc}(K_1, N_1, A_1, M_1) = \mathsf{AE.Enc}(K_2, N_2, A_2, M_2)$.

We write $\mathsf{Adv}_{\mathsf{AE}}^{\mathrm{aeadSCMT}}(\mathcal{A}; \mathsf{S})$ to denote the probability that $\mathcal{A}$ wins for the context-selector S.

Relations. Similarly, as for the MAC notions, Fig. 3 provides the relations between the different AEAD notions. The justification of these relations is also given in App. A.

Relationship to the MAC notions. Menda et al. [MLGR23] showed that for AEADs following a special format (i.e., the combination of a MAC and a NoFailDecrypt algorithm) to be context-discovery secure, a required condition is for its underlying MAC to be preimage resistant. In the following, we generalize this observation. We formalize the relationship between the committing and context-discovery security of any AEAD and its corresponding MAC function. We notably show that analyzing this corresponding MAC function only makes sense in some specific cases, with constructions loosely following Encrypt-then-MAC being one of them.

We could define the nonce-based MAC scheme $\mathsf{MAC} = (\mathsf{MAC.Tg}, \mathsf{MAC.Vf})$ corresponding to a nonce-based AEAD scheme $\mathsf{AE} = (\mathsf{AE.Enc}, \mathsf{AE.Dec})$ by

$$\mathsf{MAC.Tg}(K, N, (A, M)) = \mathsf{AE.Enc}(K, N, A, M)$$
$$\mathsf{MAC.Vf}(K, N, (A, M), T) = \begin{cases} 1 & \textbf{if } \mathsf{AE.Enc}(K, N, A, M) = T \\ 0 & \text{otherwise.} \end{cases}$$

Here, $\mathsf{MAC.Tg}$ outputs the complete output of $\mathsf{AE.Enc}$ which can be a variable length output. Obviously, analyzing the security notions defined above for the scheme AE or the scheme MAC are equivalent. Thus analyzing this MAC does not bring any advantages over analyzing the security of AE.

More interestingly, if we define instead a MAC corresponding to AE by

$$\mathsf{MAC.Tg}(K, N, (A, M)) = \mathsf{AE.Enc}(K, N, A, M)$$
$$\mathsf{MAC.Vf}(K, N, (A, M), T) = \begin{cases} 1 & \textbf{if } \mathsf{AE.Dec}(K, N, A, T) \neq \perp \\ 0 & \text{otherwise.} \end{cases}$$

Then, if we analyze the variants of the security notions defined through the functions $\mathsf{AE.Dec}$ and $\mathsf{MAC.Vf}$ (i.e. the D-notions for AEAD [BH22] and the V-notions for MAC), analyzing the above defined MAC might be simpler than analyzing AE. To observe this, we first note the equivalence between a D-notion satisfied by AE and the corresponding V-notion satisfied by MAC. Then, we remark that in the definition of $\mathsf{MAC.Vf}$, one only needs to know whether the output of $\mathsf{AE.Dec}$ is different from $\perp$ and thus does not necessarily need to decrypt the full plaintext $M = \mathsf{AE.Dec}(K, N, A, T)$. For example, in schemes following loosely Encrypt-then-MAC, it is simpler to check whether $\mathsf{AE.Dec}(K, N, A, T) \neq \perp$ than to compute $\mathsf{AE.Dec}(K, N, A, T)$.

4.3 Key-Robustness Notion for Symmetric-Key Encryption

We now show that ivE schemes cannot satisfy key-committing security but can satisfy the weaker key-robustness notion that we define. We also show that the most popular ivE satisfies this notion of key-robustness. Key-robustness will often be required to prove the security of generic composition in Sec. 6.

In the following security game, $\mathsf{SE} = (\mathsf{SE.Enc}, \mathsf{SE.Dec})$ is an IV-basedsymmetric-key encryption scheme.

Key-robustness game RBT_k for ivE. In the ivE-key-robustness game, an adversary $\mathcal{A}$ outputs (K_1, K_2, IV, M); $\mathcal{A}$ wins if:

- $K_1 \neq K_2$;
- $\mathsf{SE.Enc}(K_1, IV, M) = \mathsf{SE.Enc}(K_2, IV, M)$.

We write $\mathsf{Adv}^{\mathrm{seRBT_k}}_{\mathsf{SE}}(\mathcal{A})$ to denote the probability that $\mathcal{A}$ wins.

In the case of a block cipher, the key-robustness notion can be adapted by simply removing the IVs from the definition.

Key-Committing Insecurity of IV-Based Encryption. We could define a notion of key-committing security for IV-based encryption where an adversary $\mathcal{A}$ would break this notion if it outputs $(K_1, IV_1, M_1), (K_2, IV_2, M_2)$ such that $K_1 \neq K_2$ and $\mathsf{SE.Enc}(K_1, IV_1, M_1) = \mathsf{SE.Enc}(K_2, IV_2, M_2)$. However, as the following attack demonstrates, no IV-based encryption scheme can satisfy this notion. Consider the following attack where an adversary chooses arbitrary $(K_1, IV_1, M_1, K_2, IV_2)$ satisfying $K_1 \neq K_2$ and

```
CBC[E].Enc(K, IV, M)
1 :  $M_1 \| \cdots \| M_\ell \xleftarrow{n} M, C_0 \leftarrow IV$
2 :  for $i = 1$ to $\ell$ do
3 :      $C_i \leftarrow \mathsf{E}(K, C_{i-1} \oplus M_i)$
4 :  return $C_1 \| \cdots \| C_\ell$
```

Figure 4: CBC Encryption.

```
CTR[E].Enc(K, IV, M)
1 :  $M_1 \| \cdots \| M_\ell \xleftarrow{n} M, b \leftarrow n - |IV|$
2 :  for $i = 1$ to $\ell$ do
3 :      $C_i \leftarrow M_i \oplus \mathsf{E}(K, IV \| \langle i \rangle_b)$
4 :  return $C_1 \| \cdots \| C_\ell$
```

Figure 5: CTR Encryption.

computes $M_2 = \mathsf{SE.Dec}(K_2, IV_2, \mathsf{SE.Enc}(K_1, IV_1, M_1))$. Then by the tidiness property of SE, it holds that $\mathsf{SE.Enc}(K_1, IV_1, M_1) = \mathsf{SE.Enc}(K_2, IV_2, M_2)$ and the pair $(K_1, IV_1, M_1), (K_2, IV_2, M_2)$ breaks the key-committing security of SE.

This attack notably highlights that adapting the generic composition analyses from [FOR17] to an IV-based setting is not straightforward, as it requires the symmetric encryption scheme to be key-committing secure to prove the key-committing security of generic compositions.

Key-Robustness Security of Popular IV-Based Encryption. We now show that key-robustness is a property that can be expected from most IV-based symmetric-key encryption schemes. To do so, we analyze the key-robustness security of three popular encryption schemes: CBC [EMST78], CTR [DH79] and the Duplex [BDPV12]. The analysis for the Duplex is deferred to App. C. For CBC and CTR, we show that in the standard model, the key-robustness of these schemes can be reduced to the key-robustness security of the underlying block cipher. We also give as additional justification, an analysis of CBC and CTR in the ideal-cipher model in App. B, together with a pictorial description of the encryption procedure for these two schemes in Fig. 19 and 20.

Proposition 1. *Let* CBC[E] *and* CTR[E] *be the IV-based encryption schemes built on top of the block-cipher* E*, whose encryption procedure is described in Fig. 4 and 5. For any adversary $\mathcal{A}$ against* CBC[E] *or* CTR[E]*, there exists an adversary $\mathcal{B}$ against* E *such that*

$$\mathsf{Adv}^{\mathrm{seRBT_k}}_{\mathsf{CBC[E]/CTR[E]}}(\mathcal{A}) \leq \mathsf{Adv}^{\mathrm{seRBT_k}}_{\mathsf{E}}(\mathcal{B}).$$

Proof. For simplicity, we assume every message M to be a multiple of the block size n of the underlying block cipher E. Let $\mathcal{B}$ be the adversary that runs $\mathcal{A}$ to get an output (K_1, K_2, IV, M), and that returns $(K_1, K_2, IV \oplus \lfloor M \rfloor_n)$ in the CBC case and $(K_1, K_2, IV \| \langle 1 \rangle_{n-|IV|})$ in the CTR case. From the definition of $\mathcal{B}$, if $\mathcal{A}$ wins the ivE-key-robustness game for CBC/CTR, then $\mathcal{B}$ also wins its respective game for the block cipher E. □

5 Analyses of MAC Constructions

In this section, we show key-committing and context-committing attacks against several standardized MACs, including CBC-MAC and its variants in [ISO11a], HMAC with keys of variable length in [ISO21], Badger, Poly1305-AES and GMAC in [ISO11b], LightMAC and Chaskey in [ISO19]. These attacks can be easily adapted to context-discovery attacks against the corresponding MACs, except HMAC with variable-length keys which we cannot show context-discovery attack. We then show how to fix CBC-MAC and CMAC to achieve key-committing security that helps build AEAD from generic compositions. Finally, we show that HMAC with keys of a fixed length less than $d - 1$ bits is context-committing secure and context-discovery secure.

5.1 Attacks against block-cipher-based MACs

In this section, we show key-committing, context-committing, and context-discovery attacks against CBC-type MACs that are specified in ISO/IEC 9797-1:2011 [ISO11a]. Note that these attacks are similar to a robustness attack against $\mathsf{CBC\text{-}MAC}$ in [FOR17], while their attack is for one-block messages and our attack is for variable-length messages.

The $\mathsf{CBC\text{-}MAC}$ is built from cipher block chaining the underlying block cipher E. Let $M = m_1 || \dots || m_\ell$ be a message, where $|m_i| = n$ for $1 \le i \le \ell$ and ℓ is the block length. Then $\mathsf{CBC\text{-}MAC}_K(M)$ is defined as y_ℓ where

$$y_i = E_K(m_i \oplus y_{i-1})$$

for $1 \le i \le \ell$ and $y_0 = 0^n$.

We now present a key-committing attack against $\mathsf{CBC\text{-}MAC}$ which is specified as Algorithm 1 in ISO/IEC 9797-1:2011. A similar attack can also apply to the remaining four CBC-type MACs in ISO/IEC 9797-1:2011. The adversary first computes a tag $T = \mathsf{CBC\text{-}MAC}_{K_1}(M_1)$ for a pair (K_1, M_1). She then constructs another pair (K_2, M_2) that can produce the same tag T as follows. She chooses an arbitrary value for K_2 such that $K_2 \neq K_1$, and $\ell_2 - 1$ arbitrary values for the first $\ell_2 - 1$ blocks $m_1^2 || \dots || m_{\ell_2 - 1}^2$ of M_2. She computes the last block of M_2 as follows. She does the following computation:

$$y_i^2 = E_{K_2}(m_i^2 \oplus y_{i-1}^2)$$

for $1 \le i \le \ell_2 - 1$ and $y_0^2 = 0^n$. She computes $x_{\ell_2}^2 = E_{K_2}^{-1}(T)$. Then she obtains $m_{\ell_2}^2 = x_{\ell_2}^2 \oplus y_{\ell_2 - 1}^2$.

The context-committing attack is the same as above as it is weaker than the key-committing attack. On the other hand, the above attack can be easily adapted to a context-discovery attack against $\mathsf{CBC\text{-}MAC}$. Given any tag T, we can use a similar procedure to obtain (K_2, M_2) such that $T = \mathsf{CBC\text{-}MAC}_{K_2}(M_2)$.

Note that CBC-type MACs in ISO/IEC 9797-1:2011 can achieve standard forgery security up to $2^{n/2}$ queries due to the generic attack against single-pass CBC-MACs by Preneel and Oorschot [Pv95].

5.2 Committing attacks against dedicated-hash-based MACs

In this section, we show key-committing and context-committing attacks against HMAC [ISO21] when the key is of variable length. These attacks exploit weak key pairs in HMAC that have been used to mount key-collision attack [Stä22] and indifferentibility attack [DRST12] against HMAC.

HMAC is a hash-based MAC involving a cryptographic hash function and a secret key. Fix some hash function $H : \{0,1\}^* \to \{0,1\}^n$. Assume this hash function H is built by iterating an underlying compression function with a message block size of $d \ge n$. Given a message $M \in \{0,1\}^*$ and a key $K \in \{0,1\}^*$, the function $\mathsf{HMAC} : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^n$ is defined by

$$\mathsf{HMAC}(K, M) = H\left((K' \oplus \mathrm{opad}) || H\left((K' \oplus \mathrm{ipad}) || M\right)\right)$$

where opad and ipad are two d-bit constant strings, $K' = H(K) || 0^{d-n}$ if K is larger than the block size and $K' = K || 0^{d - |K|}$ if K is less than or equal to the block size.

Due to the key derivation, we can mount a key-committing attack against HMAC when the key is of variable length. The adversary first computes $T = \mathsf{HMAC}(K_1, M_1)$ for a pair (K_1, M_1). If $|K_1| < d$, then she sets $K_2 = K_1 || 0$. If $|K_1| > d$, then she sets $K_2 = H(K_1)$. It can be seen that the pair (K_2, M_1) will result in the same tag as the pair (K_1, M_1).

The context-committing attack is the same as above. However, we cannot use similar procedures to mount a context-discovery attack against HMAC as the above key-committing attack requires computing firstly the pair (K_1, M_1) for the tag and then exploiting the weak key pairs. While in a context-discovery attack, the adversary needs to solely start from a tag and try to find the context corresponding to this

```
Keygen(K)
1 :  $k^F_1, \dots, k^F_6 \leftarrow \mathrm{PRG}(K)$ where $k^F_1, \dots, k^F_6 \in \{0, \dots, 2^{32} - 6\}$
2 :  $k_1, \dots k_{58} \leftarrow \mathrm{PRG}(K)$ where $k_1, \dots, k_{58} \in \{0, \dots, 2^{32} - 6\}$
3 :  return $k^F_1, \dots, k^F_6, k_1, \dots, k_{58}$

Badger(K, M, N)
1 :  $\ell = |M|$
2 :  if $\ell \mod 64 \neq 0$ then $M \leftarrow M||0^{64-(\ell \mod 64)}$
3 :  for $i = 1$ to $i = \lceil \log_2(\ell/64) \rceil$ do
4 :    $M = m_1|| \dots ||m_t$ where $|m_i| = 64$
5 :    if $t$ is even then $M = H(k_i, m_1, m_2)|| \dots ||H(k_i, m_{t-1}, m_t)$
6 :                     else $M = H(k_i, m_1, m_2)|| \dots ||H(k_i, m_{t-1}, m_{t-1})||m_t$
7 :  $Q \leftarrow 0^7||\ell||M$
8 :  $Q = q_5|| \dots ||q_1$ where $|q_i| = 27$
9 :  $S = \sum_{i=1}^{5} q_i k^F_i + k^F_6 \mod (2^{32} - 5)$
10 : return $S \oplus \mathrm{PRG}(K, N)$ where $|\mathrm{PRG}(K, N)| = 32$

H(k, m_1, m_2)
1 :  return $(m_1 +_{32} k) \cdot ((m_1 \gg 32) +_{32} (k \gg 32)) +_{64} m_2$
```

Figure 6: Pseudo-code description of the Badger algorithm.

tag. The context-discovery attack against HMAC with variable-length keys may be hard as the underlying compression is typically non-invertible.

Note that the standard forgery security of HMAC has been established up to the birthday bound with keys of any sufficiently large length [BBGS23].

5.3 Attacks against universal-hash-based MACs

In this section, we present key-committing, context-committing, and context-discovery attacks against universal-hash-based MACs Badger, GMAC, Poly1305-AES that are specified in ISO/IEC 9797-3 [ISO11b]. Note that we cannot show committing attack against the remaining one UMAC in ISO/IEC 9797-3 and thus its committing security is still open.

Attacks against Badger. Badger is a universal-hash-based MAC designed by Boesgaard [BCZ05] and standardized in [ISO21]. It follows the Wegman-Carter framework [WC81], and employs a binary-tree method to improve the efficiency of hashing. A pseudorandom generator is used in the initialization to generate random keys for hashing. These random keys can be reused for the coming messages. This pseudorandom generator is also used in the finalization with a nonce to generate the random string to mask the hash value. Given a key K, a nonce N, and a message M, it outputs a 32-bit tag as $T = \mathsf{Badger}_K(N, M)$. The pseudo-code definition of Badger is illustrated in Fig. 6.

We mount a key-committing attack against Badger as follows. The adversary first computes the tag $T = \mathsf{Badger}_{K_1}(N_1, M_1)$ for a tuple of (K_1, N_1, M_1). She then can construct a different tuple of (K_2, N_2, M_2)

such that $(K_2, N_2, M_2) \neq (K_1, N_1, M_1)$ but can result in the same tag value T as follows. Here K_2 and N_2 can be arbitrarily chosen and $M_2 = m_1||m_2$ is a 128-bit string crafted by the adversary. The adversary computes $S_2 = T \oplus \mathrm{PRG}(K_2, N_2)$ for arbitrary K_2 and N_2. She solves the equation $S_2 = \sum_{i=1}^{5} q_i k_i^F + k_6^F \mod (2^{32} - 5)$ in line 9 to obtain one possible value for M in line 7 of the function $\mathsf{Badger}(K, M, N)$ of Fig. 6. By fixing m_1, she solves the equation in line 1 of the function $H(k, m_1, m_2)$ to obtain m_2. The context-committing attack is the same as above.

The context-committing attack is the same as the above key-committing attack. On the other hand, given a tag T, we can use the similar procedures to find the tuple (K_2, N_2, M_2) such that $T = \mathsf{Badger}_{K_2}(N_2, M_2)$, there with mounting the context-discovery attack against Badger.

Note that the standard forgery probability for Badger is around $2^{-26.12}$ for 32-bit tag and can be smaller if we allow a longer tag as discussed in [BCZ05].

Attacks against $\mathsf{Poly1305\text{-}AES}$. $\mathsf{Poly1305\text{-}AES}$ [Ber05,ISO21] is a universal-hash-based MAC following the Wegman-Carter-Shoup framework [Sho96], using polynomial evaluation modulo the prime number $2^{130} - 5$ for better performance. It uses the AES with 128-bit key K, and the hash key r has 128-bit, 22 bits of which are set to 0. Given a message $M = m_1|| \ldots ||m_\ell$ where $|m_i| = 128$ for $1 \leq i \leq \ell - 1$ and $0 < |m_\ell| \leq 128$, each message block is first padded to 129-bit value c_i. Then the MAC of a ℓ-block message M with nonce N is defined as:

$$\begin{aligned} &\mathsf{Poly1305\text{-}AES}_{K,r}(M, N) = \mathsf{Poly1305}_r(M) +_{128} \mathsf{AES}_K(N) \\ =&(((c_1 r^\ell + c_2 r^{\ell-1} + \ldots + c_\ell r) \mod 2^{130} - 5) + \mathsf{AES}_K(N)) \mod 2^{128}. \end{aligned} \quad (1)$$

We mount a key-committing attack against $\mathsf{Poly1305\text{-}AES}$ as follows. The adversary first computes a tag $T = \mathsf{Poly1305\text{-}AES}_{K_1,r_1}(M_1, N_1)$ for a tuple of (K_1, r_1, M_1, N_1). She can then craft another different tuple of (K_2, r_2, M_2, N_2) with the same tag T such that $(K_2, r_2, M_2, N_2) \neq (K_1, r_1, M_1, N_1)$. She chooses arbitrary values for K_2, r_2, M_2. She solves the following equation

$$N_2 = \mathsf{AES}_{K_2}^{-1}\left(\mathsf{Poly1305\text{-}AES}_{K_1,r_1}(M_1, N_1) -_{128} \mathsf{Poly1305}_{r_2}(M_2)\right)$$

to obtain N_2. Then the tuple (K_2, r_2, M_2, N_2) will result in the same tag value T. This attack is better than the one against $\mathsf{ChaCha\text{-}Poly1305}$ [LGR20] as the previous work only works for about one-quarter of possible inputs while we don't have this restriction here.

The context-committing attack is the same as above. On the other hand, given a tag T, we can use the similar procedures to find the tuple (K_2, r_2, M_2, N_2) such that $T = \mathsf{Poly1305\text{-}AES}_{K_2,r_2}(M_2, N_2)$, therewith mounting context-discovery attack against $\mathsf{Poly1305\text{-}AES}$.

Note that the standard forgery security of $\mathsf{Poly1305\text{-}AES}$ is up to the birthday bound 2^{64} as shown in [Ber05, Nan18].

Attacks against GMAC. GMAC [MV04,ISO21,Dwo07a] is an authentication mode following the Wegman-Carter-Shoup framework with the Galois Hash function (GHASH) and a block cipher. The GHASH function evaluates a Galois Field polynomial at the hash key H, whose coefficients are the message blocks and block length. The output of this hash function is then xored with the output of the block cipher with a nonce to generate the tag. Given a message $M = m_1|| \ldots ||m_\ell$, the output of GMAC is defined as:

$$\mathsf{GMAC}_K(M, N) = \left(m_1 H^{\ell+1} \oplus \ldots \oplus m_\ell||0^* H^2 \oplus |M|_n H\right) \oplus E_K(N\|0^{31}\|1)$$

where $H = E_K(0^n)$.

We show a key-committing attack against GMAC as follows. The adversary first computes a tag $T = \mathsf{GMAC}_{K_1}(M_1, N_1)$ for a tuple of (K_1, M_1, N_1). She then constructs another different tuple of (K_2, M_2, N_2)

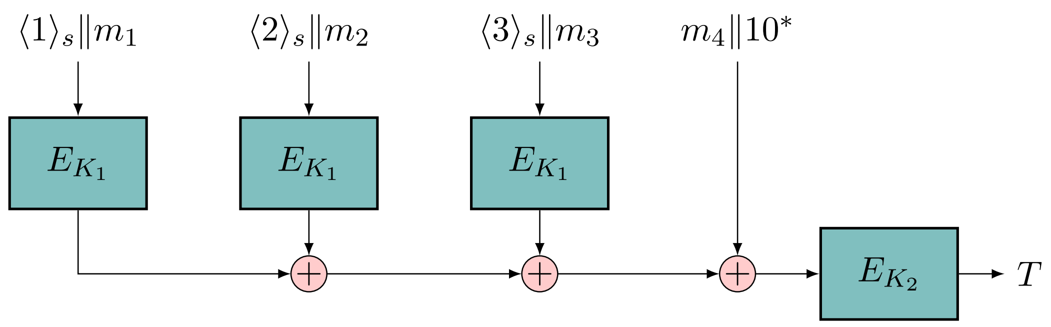
$\langle 1\rangle_s \| m_1$
$\langle 2\rangle_s \| m_2$
$\langle 3\rangle_s \| m_3$
$m_4 \| 10^*$
E_{K_1}
E_{K_1}
E_{K_1}
E_{K_2}
T

Figure 7: The LightMAC construction for a message $m_1||m_2||m_3||m_4 \xleftarrow{n-s} M$.

that can result in the tag T. She chooses arbitrary values for K_2 and N_2, and fixes the first $\ell' - 1$ blocks of M_2. She then solves the following equation to obtain the last block $m'_{\ell'}$ of M_2:

$$T = \left(m'_1 H_2^{\ell+1} \oplus \ldots \oplus m'_{\ell'} || 0^* H_2^2 \oplus |M_2|_n H \right) \oplus E_{K_2}(N_2).$$

The context-committing attack is the same as above. On the other hand, given a tag T, we can use similar procedures to find the tuple (K_2, M_2, N_2) such that $T = \mathsf{GMAC}_{K_2}(M_2, N_2)$, therewith mounting context-discovery attack against GMAC.

Note that the standard forgery security of GMAC is up to the birthday bound $2^{n/2}$ [MV04, IOM12].

5.4 Attacks against lightweight MACs

In this section, we show key-committing, context-committing, and context-discovery attacks against lightweight MACs including LightMAC and Chaskey that are standardized in ISO/IEC 29192-6:2019 [ISO19]. Note that we cannot find a committing attack against Tsudik's keymode in ISO/IEC 29192-6:2019 and thus its committing security is still open.

Attacks against LightMAC. LightMAC is designed by Luykx et al. [LPTY16] for lightweight block ciphers and standardized in ISO/IEC 29192-6:2019 [ISO19]. Its provable security bound $q^2/2^n$ is independent of the message length. For authentication, a message M will be first split into $(n-s)$-bit blocks with the length of the last block being anywhere from zero to $n-s-1$ bits. Each of the first $\ell - 1$ blocks is concatenated with a s-bit counter to become a full n-bit block, and the last block is padded with 10^* to become a full n-bit block. See Fig. 7 for an illustration of LightMAC.

We then mount a key-committing attack against LightMAC with success probability 1 when the message length is less than $n-s-1$, and another context-committing attack against LightMAC with success probability 2^{-s} for longer messages. We begin with the first attack for short messages. The adversary first computes a tag $T = \mathsf{LightMAC}_{K_1,K_2}(M_1) = E_{K_2}(M_1 \| 10^*)$ for a tuple (K_1, K_2, M_1) where $|M_1| \le n-s-1$. She then constructs another different tuple (K'_1, K_2, M_1) where $K'_1 \ne K_1$. Obviously, this tuple will result in the same tag T as $T = E_{K_2}(M_1 \| 10^*)$.

We then mount another key-committing attack against LightMAC with success probability 2^{-s-1} for long messages. The adversary first computes a tag $T = \mathsf{LightMAC}_{K_1,K_2}(M_1)$ for a tuple (K_1, K_2, M_1). She then constructs another different tuple (K'_1, K'_2, M_2) that will result in the same tag T with probability 2^{-s}. She chooses two arbitrary values for K'_1 and K'_2, and computes $Y = E^{-1}_{K'_2}(T)$. She fixes the first $\ell_2 - 1$ blocks $m_1^2 || \ldots || m^2_{\ell_2 - 1}$ of M_2. Then she computes $X = Y \oplus E_{K'_1}(\langle 1\rangle_s \| m_1^2) \oplus \ldots \oplus E_{K'_1}(\langle \ell_2 - 1\rangle_s \| m^2_{\ell_2-1})$. If the string X is in the form of $x || 10^*$ where $0 \le |x| \le n-s-1$, then she can fix the value of the last

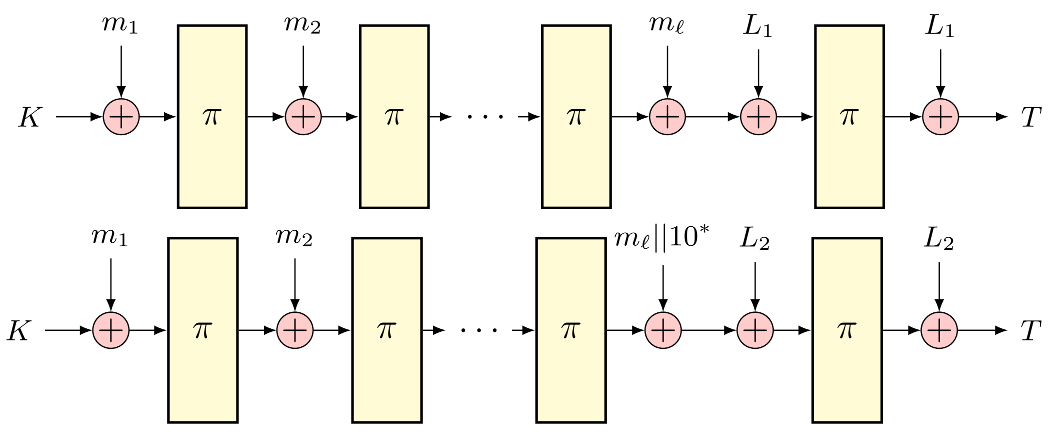

Figure 8: The Chaskey construction when $|m_\ell| = n$ (top), and when $0 < |m_\ell| < n$. The subkeys L_1 and L_2 are derived from the master key K.

block of M_2 to x, which happens with probability around 2^{-s-1}. If the string X is not in this form, then this attack will fail.

These two key-committing attacks also apply in the context-committing setting. On the other hand, given a tag T, we can use the similar procedures to obtain (K_1', K_2', M_2) such that $T = \mathsf{LightMAC}_{K_1', K_2'}(M_2)$ with the same probability. Thus, we can also mount a context-discovery attack against LightMAC.

Note that the standard forgery security of LightMAC is up to the birthday bound $2^{n/2}$ [LPTY16,CJN21].

Remark. Similar committing attacks for long messages can be applied to PMAC [Rog04] but with success probability 1 as the length of the padding of PMAC can be 0.

Attacks against Chaskey. Chaskey [MMV$^+$14] is a permutation-based MAC that is designed to be extremely fast on 32-bit microcontrollers. It can also be described as CBC-MAC with an Even-Mansour cipher. A pictorial illustration of Chaskey is given in Fig. 8.

We show a key-committing attack against Chaskey when messages are of full blocks and the tag is of n bits as follows. The adversary first computes a tag $T = \mathsf{Chaskey}_{K_1}(M_1)$ for a pair (K_1, M_1). She then constructs another different pair (K_2, M_2) that can result in the same tag T. She chooses an arbitrary value for K_2, and computes the subkeys L_1^2 and L_2^2. By fixing the first $\ell_2 - 1$ blocks of M_2, she can invert from T to obtain the last block of M_2 as she can compute all the internal values by herself.

The context-committing attack is the same as above. We can also use similar procedures to find a pair (K_2, M_2) such that $T = \mathsf{Chaskey}_{K_2}(M_2)$ for a given tag T. Hence, we can mount a context-discovery attack against Chaskey.

Note that the standard forgery security of Chaskey is up to the birthday bound $2^{n/2}$ [MMV$^+$14].

5.5 Fixing MACs for key-committing security

In this section, we show how to fix CBC-MAC and CMAC in [ISO11a] to be key-committing secure and context-discovery secure. The main idea is to replace the last block cipher call in these two schemes with the Davies-Meyer construction. By doing so, the adversary needs to find a collision against the Davies-Meyer construction to mount the key-committing attack, which is similar in spirit to the fixes for GCM and GCM-SIV in [BH22]. Furthermore, regarding our newly proposed definition of context-discovery security, we show that context-discovery security can also be reduced to the preimage resistance of the Davies-Meyer construction.

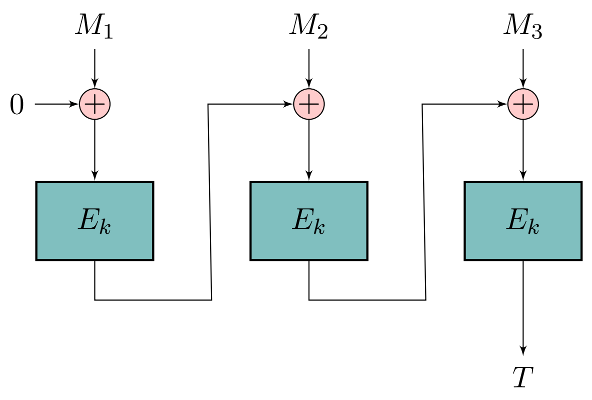

Figure 9: The CBC-MAC construction for a message $M = m_1||m_2||m_3$.

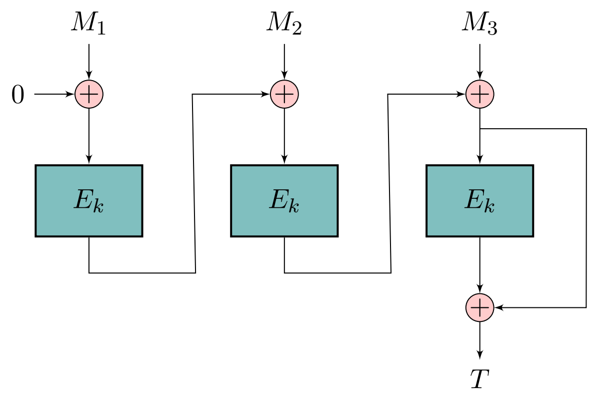

Figure 10: The CBC-MAC-C1 construction for a message $M = m_1||m_2||m_3$.

An efficient key-committing secure variant of CBC-MAC. CBC-MAC-C1 is a variant of CBC-MAC by replacing the last block cipher call in CBC-MAC with the Davies-Meyer construction to achieve key-committing security. It enjoys almost the same efficiency and is a secure MAC for prefix-free or fixed-length messages as CBC-MAC. The illustrations of the original CBC-MAC and CBC-MAC-C1 can be found in Fig. 9 and Fig. 10. In the figures the function derive : $\{0,1\}^n \to \{0,1\}^n$ is defined as follows,

$$\text{derive}(X) := \begin{cases} X << 1 & \text{if } \lceil X \rceil_1 = 0 \\ (X << 1) \oplus R_n & \text{otherwise} \end{cases},$$

for some constant R_n solely dependent on the value of n.

We first recall the Davies-Meyer construction DM and its collision-resistant security which is helpful in the later analyses. Let $E : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ be a block cipher. The Davies-Meyer construction $\mathsf{DM} : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ is defined by

$$\mathsf{DM}[E](K, X) = E_K(X) \oplus X.$$

The following lemma shows that DM is collision resistance and everywhere preimage resistance [RS04].

Lemma 1. *For any adversary that makes at most $p \le 2^{n-1}$ ideal-cipher queries to E and E^{-1}, we have*

$$\mathsf{Adv}^{\text{coll}}_{\mathsf{DM}}(\mathcal{A}) \le \frac{p^2}{2^n}, \qquad \mathsf{Adv}^{\text{epre}}_{\mathsf{DM}}(\mathcal{A}) \le \frac{2p}{2^n}.$$

For completeness, the proof is recalled in App. D.

The following theorems show that CBC-MAC-C1 can ensure key-committing security and context-discovery security in the ideal-cipher model. The ideal-cipher model is natural for the key-committing setting as in this setting, the adversary can choose the key of the underlying block cipher for queries.

Theorem 1. *Suppose that the context-selector has access to neither CBC-MAC-C1 nor its underlying block cipher. Then for any adversary $\mathcal{A}$ that makes at most $p \le 2^{n-1}$ ideal-cipher queries, we have*

$$\mathsf{Adv}^{\text{macCMT}_\text{k}}_{\textsf{CBC-MAC-C1}}(\mathcal{A}) \le \frac{p^2}{2^n}, \qquad \mathsf{Adv}^{\text{macCDY}}_{\textsf{CBC-MAC-C1}}(\mathcal{A}) \le \frac{2p}{2^n}.$$

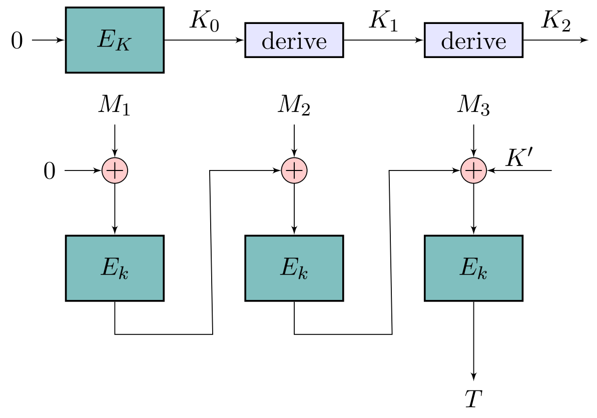

Figure 11: The CMAC construction for a message $M = m_1||m_2||m_3$. Here, $K' = \begin{cases} K_1 & \text{if } |M_3| = n \\ K_2 & \text{otherwise.} \end{cases}$

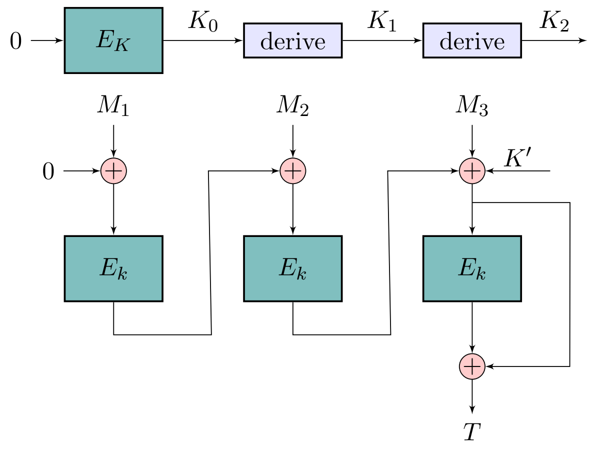

Figure 12: The CMAC-C1 construction for a message $M = m_1||m_2||m_3$. Here, $K' = \begin{cases} K_1 & \text{if } |M_3| = n \\ K_2 & \text{otherwise.} \end{cases}$

Proof. Suppose that the adversary $\mathcal{A}$ wins the key-committing game as defined in Sec. 4, namely she finds (K_1, M_1) and (K_2, M_2) such that $\textsf{CBC-MAC-C1}_{K_1}(M_1) = \textsf{CBC-MAC-C1}_{K_2}(M_2)$ and $K_1 \neq K_2$, then we can construct another adversary $\mathcal{B}$ against the collision-resistant security of the Davies-Meyer construction DM. Let X_1 and X_2 be the inputs to the Davies-Meyer construction in CBC-MAC-C1 for the pairs (K_1, M_1) and (K_2, M_2) respectively. Then (X_1, K_1) and (X_2, K_2) will be a valid collision pair against DM as $K_1 \neq K_2$ if the adversary $\mathcal{A}$ wins the key-committing game. Hence, $\textsf{Adv}^{\text{macCMT}_\text{k}}_{\textsf{CBC-MAC-C1}}(\mathcal{A}) \leq \textsf{Adv}^{\text{coll}}_{\textsf{DM}}(\mathcal{B}) \leq \frac{p^2}{2^n}$ from Lem. 1.

Suppose that the adversary wins the context-discovery game against CBC-MAC-C1 as defined in Sec. 4.1, namely given a challenge tag T by the context-selector S, she outputs (K, N, M) such that $\textsf{CBC-MAC-C1}_K(M) = T$, then we can construct another adversary $\mathcal{B}$ against the everywhere preiamge resistance of the Davies-Meyer construction DM as follows. Let X be the input to the Davies-Meyer construction in CBC-MAC-C1 for the tuple (K, N, M). Then X will be a valid preimage against DM for the image T if the adversary $\mathcal{A}$ wins the context-discovery game. Hence, $\textsf{Adv}^{\text{macCDY}}_{\textsf{CBC-MAC-C1}}(\mathcal{A}) \leq \textsf{Adv}^{\text{epre}}_{\textsf{DM}}(\mathcal{A}) \leq \frac{2p}{2^n}$ by assuming $p \leq 2^{n-1}$. □

Note that the previous context-committing attack against CBC-MAC with $K_1 = K_2$ can still work for CBC-MAC-C1, but the key-committing attack with the restriction that $K_1 \neq K_2$ and the context-discovery attack cannot work anymore due to the collision resistance and preimage resistance of the Davies-Meyer construction.

The PRF security of CBC-MAC-C1 for prefix-free messages is formulated by the following theorem. The proof is similar to the proof for CBC-MAC [BPR05, JN16], and thus omitted in this paper.

Theorem 2. *For any adversary $\mathcal{A}$ that makes at most q prefix-free queries, each query being at most ℓ blocks, and the total number of blocks of all queries being at most σ, we have*

$$\textsf{Adv}^{\text{prf}}_{\textsf{CBC-MAC-C1}}(\mathcal{A}) \leq \frac{8\sigma q}{2^n} + \frac{8\sigma q \ell^3}{2^{2n}} + \frac{q^2}{2^{n+1}}.$$

An efficient key-committing secure variant of CMAC. Following the similar idea in CBC-MAC-C1, we proposed a key-committing secure variant of CMAC called CMAC-C1 by replacing the last block cipher call in CMAC with the Davies-Meyer construction. It is almost efficient and is a secure MAC for variable-length messages as CMAC. The illustrations of CMAC and CMAC-C1 are given in Fig. 11 and Fig. 12 respectively.

The following theorems show that CMAC-C1 can provide key-committing security and context-discovery security in the ideal-cipher model. The proof is similar to the one for Thm 1 as we can reduce the key-committing security and context-discovery security of CMAC-C1 to the collision resistance and the preimage resistance of the underlying DM construction.

Theorem 3. *Suppose that the context-selector has access to neither CMAC-C1 nor its underlying block cipher. Then for any adversary that makes at most $p \leq 2^{n-1}$ ideal-cipher queries, we have*

$$\mathsf{Adv}^{\mathrm{macCMT_k}}_{\textsf{CMAC-C1}}(\mathcal{A}) \leq \frac{p^2}{2^n}, \qquad \mathsf{Adv}^{\mathrm{macCDY}}_{\textsf{CMAC-C1}}(\mathcal{A}) \leq \frac{2p}{2^n}.$$

Similar as CBC-MAC-C1, the previous context-committing attack against CBC-MAC can still work but the key-committing attack and the context-discovery attack cannot work due to the collision resistance preimage resistance of the Davies-Meyer construction. Hence, CMAC-C1 can be used in applications to ensure data authenticity when key-committing and context-discovery attacks are relevant, and cannot be used when the context-committing attack is relevant. How to fix CMAC to achieve the stronger context-committing security is left as an interesting future work.

The PRF security of CMAC-C1 is captured by the following theorem. The proof is similar to the previous proofs for CMAC [Nan09, CJN22, IK03], and thus is omitted in this paper.

Theorem 4. *For any adversary $\mathcal{A}$ that makes at most q queries, each query being at most ℓ blocks, and the total number of blocks of all queries being at most σ, we have*

$$\mathsf{Adv}^{\mathrm{prf}}_{\textsf{CMAC-C1}}(\mathcal{A}) \leq \frac{5\sigma q}{2^n} + \frac{8q^2\ell^4}{2^{2n}} + \frac{q^2}{2^{n+1}}.$$

5.6 Context-committing security of HMAC

In this section, we show that HMAC with keys of a fixed length less than $d-1$ bits is context-committing secure and context-discovery secure.

In [DRST12, Theorem 4.4], the authors proved that HMAC is indifferentiable from a random oracle when the key is of a fixed length less than $d-1$ bits. In the context-committing setting, if the adversary can find a pair (K_1, M_1) and (K_2, M_2) such that $(K_1, M_1) \neq (K_2, M_2)$ and $\mathsf{HMAC}(K_1, M_1) = \mathsf{HMAC}(K_2, M_2)$, then we can use this pair to attack the indifferentiability of HMAC. Hence, the context-committing security of HMAC can be obtained from its indifferentiable security which is stated by the following theorem. The context-discovery security of HMAC with fixed-length keys can be obtained similarly. We refer the indifferentiability proof to [DRST12].

Theorem 5. *Fix $d, n > 0$. Let $H : \{0,1\}^* \to \{0,1\}^n$ be a hash function that is modeled as a random oracle. Suppose that the key of HMAC is fixed and less than $d-1$ bits, and the context-selector S does not access to neither HMAC nor H. Then for any adversary $\mathcal{A}$ that makes at most σ queries to H, we have*

$$\mathsf{Adv}^{\mathrm{macCMT}}_{\textsf{HMAC}}(\mathcal{A}) \leq \frac{2\sigma^2}{2^n}, \qquad \mathsf{Adv}^{\mathrm{macCDY}}_{\textsf{HMAC}}(\mathcal{A}; \mathsf{S}) \leq \frac{2\sigma^2}{2^n}.$$

Remark. In [DRST12], the authors also showed how to instantiate the hash function H with the Merkle-Damgard transform and proved that this instantiation can achieve a similar indifferentiable bound.

```
$\mathsf{AE}_1^{\mathsf{EaM}}[\mathsf{SE},\mathsf{MAC}].\mathsf{Enc}(K,N,A,M)$
1: $IV \leftarrow N$
2: $(K_e,K_m) \leftarrow K$
3: $C \leftarrow \mathsf{SE.Enc}(K_e,IV,M)$
4: $M^* \leftarrow \mathsf{pad}(A,M)$
5: $T \leftarrow \mathsf{MAC.Tg}(K_m,N,M^*)$
6: $\textbf{return } (C,T)$
```

```
$\mathsf{AE}_2^{\mathsf{EaM}}[\mathsf{SE},\mathsf{MAC}].\mathsf{Enc}(K,N,A,M)$
1: $A^* \leftarrow \mathsf{pad}(A)$
2: $IV \leftarrow \mathsf{MAC.Tg}(K_m,N,A^*)$
3: $(K_e,K_m) \leftarrow K$
4: $C \leftarrow \mathsf{SE.Enc}(K_e,IV,M)$
5: $M^* \leftarrow \mathsf{pad}(A,M)$
6: $T \leftarrow \mathsf{MAC.Tg}(K_m,N,M^*)$
7: $\textbf{return } (C,T)$
```

Figure 13: Encrypt-and-MAC encryption: Variant 1 (left) and Variant 2 (right).

```
$\mathsf{AE}_1^{\mathsf{EtM}}[\mathsf{SE},\mathsf{MAC}].\mathsf{Enc}(K,N,A,M)$
1: $IV \leftarrow N$
2: $(K_e,K_m) \leftarrow K$
3: $C \leftarrow \mathsf{SE.Enc}(K_e,IV,M)$
4: $C^* \leftarrow \mathsf{pad}(A,C)$
5: $T \leftarrow \mathsf{MAC.Tg}(K_m,N,C^*)$
6: $\textbf{return } (C,T)$
```

```
$\mathsf{AE}_2^{\mathsf{EtM}}[\mathsf{SE},\mathsf{MAC}].\mathsf{Enc}(K,N,A,M)$
1: $A^* \leftarrow \mathsf{pad}(A)$
2: $IV \leftarrow \mathsf{MAC.Tg}(K_m,N,A^*)$
3: $(K_e,K_m) \leftarrow K$
4: $C \leftarrow \mathsf{SE.Enc}(K_e,IV,M)$
5: $C^* \leftarrow \mathsf{pad}(A,C)$
6: $T \leftarrow \mathsf{MAC.Tg}(K_m,N,C^*)$
7: $\textbf{return } (C,T)$
```

Figure 14: Encrypt-then-MAC encryption: Variant 1 (left) and Variant 2 (right).

6 Revisiting the Security of Generic Composition

Having shown in the previous sections that some MACs are committing/context discovery secure and that popular ivE such as CTR/CBC are key-robust, we can now use these results to build committing/context discovery secure AEADs through generic composition. To be able to do so, we revisit generic composition through the lens of committing and context-discovery security. The different nuances in the security notions we define in Sec. 4 allow us to obtain new insights into these paradigms that contradict previous beliefs. For example, Menda et al. [MLGR23] gave an attack on the SIV scheme standardized in RFC 5297. Still, we show below that if SIV is instantiated with different primitives, committing and context-discovery security are attainable. We note that compared to [MLGR23], which mostly defined and used context-discovery to break the committing security of practical AEAD schemes, we provide in this section the first provable way of building a context-discovery secure AEAD.

6.1 Generic Composition Paradigms

Bellare and Namprempre [BN00] were the first to formally study the generic composition approach, which consists of combining an encryption scheme with a MAC to build a secure authenticated encryption scheme. In their paper, they showed that if the encryption scheme is assumed to be probabilistic, then encrypt-then-mac is the only secure paradigm. Namprempre et al. [NRS14] revisited the approach by assuming the encryption scheme to be either IV or nonce based, which is closer to practice. They studied all possible combinations with a MAC to build a secure nonce-based AEAD. Nine modes constructed from IV-based encryption (A1-A9) and three modes constructed from nonce-based encryption (N1-N3) were

```
$\mathsf{AE}_1^{\mathsf{MtE}}[\mathsf{SE},\mathsf{MAC}].\mathsf{Enc}(K,N,A,M)$
1 : $IV \leftarrow N$
2 : $(K_e, K_m) \leftarrow K$
3 : $M^* \leftarrow \mathsf{pad}(A,M)$
4 : $T \leftarrow \mathsf{MAC.Tg}(K_m,N,M^*)$
5 : $M^\dagger \leftarrow M||T$
6 : $C^\dagger \leftarrow \mathsf{SE.Enc}(K_e,IV,M^\dagger)$
7 : return $C^\dagger$
```

```
$\mathsf{AE}_2^{\mathsf{MtE}}[\mathsf{SE},\mathsf{MAC}].\mathsf{Enc}(K,N,A,M)$
1 : $A^* \leftarrow \mathsf{pad}(A)$
2 : $IV \leftarrow \mathsf{MAC.Tg}(K_m,N,A^*)$
3 : $(K_e, K_m) \leftarrow K$
4 : $M^* \leftarrow \mathsf{pad}(A,M)$
5 : $T \leftarrow \mathsf{MAC.Tg}(K_m,N,M^*)$
6 : $M^\dagger \leftarrow M||T$
7 : $C^\dagger \leftarrow \mathsf{SE.Enc}(K_e,IV,M^\dagger)$
8 : return $C^\dagger$
```

Figure 15: MAC-then-Encrypt encryption: Variant 1 (left) and Variant 2 (right).

```
$\mathsf{AE}^{\mathsf{SIV}}[\mathsf{SE},\mathsf{MAC}].\mathsf{Enc}(K,N,A,M)$
1 : $(K_e, K_m) \leftarrow K$
2 : $M^* \leftarrow \mathsf{pad}(A,M)$
3 : $T \leftarrow \mathsf{MAC.Tg}(K_m,N,M^*)$
4 : $IV \leftarrow T$
5 : $C \leftarrow \mathsf{SE.Enc}(K_e,IV,M)$
6 : return $(C,T)$
```

```
$\mathsf{AE}^{\mathsf{SIV}}[\mathsf{SE},\mathsf{MAC}].\mathsf{Dec}(K,N,A,C,T)$
1 : $IV \leftarrow T$
2 : $(K_e, K_m) \leftarrow K$
3 : $M \leftarrow \mathsf{SE.Dec}(K_e,IV,C)$
4 : $M^* \leftarrow \mathsf{pad}(A,M)$
5 : if $\mathsf{MAC.Vf}(K_m,N,M^*,T) = 1$
6 :   return $M$
7 : else
8 :   return $\perp$
9 : endif
```

Figure 16: SIV encryption and decryption functions.

proven secure. The security of four modes (A10-A12, N4) remained undecided and were later proved to be insecure [BPP18, Ber23]. In the following, we revisit the security analysis of the schemes from Namprempre et al. focusing mainly on the schemes N1, N2, N3, and some of the A schemes.

We first describe the generic compositions we consider in this paper, built from an IV-based encryption $\mathsf{SE} = (\mathsf{SE.Enc}, \mathsf{SE.Dec})$ and a message authentication code $\mathsf{MAC} = (\mathsf{MAC.Tg}, \mathsf{MAC.Vf})$. In all the constructions, independent keys K_e for SE and K_m for MAC together constitute the key K for AE. We also consider key-schedule-based versions of some of the constructions using a collision-resistant key-schedule KS. We assume pad to be an injective padding function.

While [NRS14] considered only PRF secure vector-input MACs as building blocks, we slightly change the syntax of the MAC for versatility, so that it can be instantiated either with a nonce-based MAC or a vector-input MAC. This change is minimal and AEAD security of the composition is maintained. We split our analyzed schemes into four groups according to their construction: Encrypt-and-MAC, Encrypt-then-MAC, MAC-then-Encrypt, and SIV. The analysis for schemes of the same group will often be similar.

Encrypt-and-MAC. In the Encrypt-and-MAC composition paradigm, the ciphertext C and the tag T are computed parallelly using calls to $\mathsf{SE.Enc}$ and $\mathsf{MAC.Tg}$ respectively. In variant 1 (Fig. 13, left), the nonce is used directly as IV for SE, and in variant 2 (Fig. 13, right), the IV is computed through a call to $\mathsf{MAC.Tg}$. We denote these as $\mathsf{AE}_1^{\mathsf{EaM}}[\mathsf{SE},\mathsf{MAC}]$ and $\mathsf{AE}_2^{\mathsf{EaM}}[\mathsf{SE},\mathsf{MAC}]$ respectively, and write $\mathsf{AE}^{\mathsf{EaM}}[\mathsf{SE},\mathsf{MAC}]$ to indicate either. These schemes correspond to the schemes N1 and A2 from [NRS14].

```
kAE[KS, AE, SE, MAC].Enc(K, N, A, M)
1 :  $(K_e, K_m) \leftarrow \mathsf{KS}(K)$
2 :  $(C, T) \leftarrow \mathsf{AE}[\mathsf{SE}, \mathsf{MAC}].\mathsf{Enc}(K' := (K_e, K_m), N, A, M)$
3 :  return $(C, T)$

kAE[KS, AE, SE, MAC].Dec(K, N, A, C, T)
1 :  $(K_e, K_m) \leftarrow \mathsf{KS}(K)$
2 :  $X \leftarrow \mathsf{AE}[\mathsf{SE}, \mathsf{MAC}].\mathsf{Dec}(K' := (K_e, K_m), N, A, C, T)$
3 :  return $X$
```

Figure 17: Key-schedule-based Authenticated Encryption.

Encrypt-then-MAC. In the Encrypt-then-MAC composition paradigm, the ciphertext C is computed first through $\mathsf{SE.Enc}$, and the tag T is then computed using C as an input to $\mathsf{MAC.Tg}$. As before, in variant 1 (Fig. 14, left), the nonce is used directly as IV for SE, and in variant 2 (Fig. 14, right), the IV is computed through a call to $\mathsf{MAC.Tg}$. We denote these as $\mathsf{AE}_1^{\mathsf{EtM}}[\mathsf{SE}, \mathsf{MAC}]$ and $\mathsf{AE}_2^{\mathsf{EtM}}[\mathsf{SE}, \mathsf{MAC}]$ respectively. These schemes correspond to the schemes N2 and A6 from [NRS14].

MAC-then-Encrypt. In the MAC-then-Encrypt composition paradigm, and intermediate tag is first computed using $\mathsf{MAC.Tg}$, and appended to the message to form a longer message $M^\dagger$, which is then encrypted through $\mathsf{SE.Enc}$ to obtain a long ciphertext $C^\dagger$. Again, In variant 1 (Fig. 15, left), the nonce is used directly as IV for SE, and in variant 2 (Fig. 15, right), the IV is computed through a call to $\mathsf{MAC.Tg}$. We denote these as $\mathsf{AE}_1^{\mathsf{MtE}}[\mathsf{SE}, \mathsf{MAC}]$ and $\mathsf{AE}_2^{\mathsf{MtE}}[\mathsf{SE}, \mathsf{MAC}]$ respectively. These schemes correspond to the schemes N3 and A8 from [NRS14].

SIV. In the SIV composition paradigm (Fig. 16), the tag T is first computed using $\mathsf{MAC.Tg}$; then C is computed using T as IV in $\mathsf{SE.Enc}$. We denote this as $\mathsf{AE}^{\mathsf{SIV}}[\mathsf{SE}, \mathsf{MAC}]$. This scheme corresponds to the scheme A4 from [NRS14].

AEAD with a Key Schedule. For each of the composition paradigms above, we will also consider a KS-based variant, which uses a Key Schedule function KS to generate the MAC key K_m and the ivE key K_e. We will model the key-schedule as a function $\mathsf{KS} : \mathcal{K} \to \mathcal{K}_e \times \mathcal{K}_m$, that takes a master key K as input and outputs a pair of keys (K_e, K_m). If $\mathsf{AE}[\mathsf{SE}, \mathsf{MAC}]$ is an AEAD scheme which internally calls SE and MAC, then we will denote its key-schedule-based version as $\mathsf{kAE}[\mathsf{KS}, \mathsf{AE}, \mathsf{SE}, \mathsf{MAC}]$. For the key-schedule-based variants of the four composition paradigms above, we'll use the following shorthands:

$$\begin{aligned}
\mathsf{kAE}_i^{\mathsf{EaM}}[\mathsf{KS}, \mathsf{SE}, \mathsf{Enc}] &:= \mathsf{kAE}[\mathsf{KS}, \mathsf{AE}_i^{\mathsf{EaM}}, \mathsf{SE}, \mathsf{Enc}], && i \in \{1, 2\},\\
\mathsf{kAE}_i^{\mathsf{EtM}}[\mathsf{KS}, \mathsf{SE}, \mathsf{Enc}] &:= \mathsf{kAE}[\mathsf{KS}, \mathsf{AE}_i^{\mathsf{EtM}}, \mathsf{SE}, \mathsf{Enc}], && i \in \{1, 2\},\\
\mathsf{kAE}_i^{\mathsf{MtE}}[\mathsf{KS}, \mathsf{SE}, \mathsf{Enc}] &:= \mathsf{kAE}[\mathsf{KS}, \mathsf{AE}_i^{\mathsf{MtE}}, \mathsf{SE}, \mathsf{Enc}], && i \in \{1, 2\},\\
\mathsf{kAE}^{\mathsf{SIV}}[\mathsf{KS}, \mathsf{SE}, \mathsf{Enc}] &:= \mathsf{kAE}[\mathsf{KS}, \mathsf{AE}^{\mathsf{SIV}}, \mathsf{SE}, \mathsf{Enc}].
\end{aligned}$$

In our proofs, we will require KS to be collision-resistant on its second output, but will not require any cryptographic hardness from it. Thus, for any pair of functions $g_e : \mathcal{K} \to \mathcal{K}_e$ and $g_m : \mathcal{K} \to \mathcal{K}_m$, $\mathsf{KS}(K) := (g_e(K), g_m(K))$ will be a valid key-schedule as long as g_m is injective. However, any cryptographic key-derivation function or KDF that is collision-resistant can also be used as the key-schedule. In practice,

Scheme	Assumption	SCMT	CDY	CDY_k	CMT	CMT_k
EaM	none	?	CDY	CDY_k	?	?
EaM	RBT_k	SCMT	CDY	CDY_k	CMT	?
KEaM	COLL	?	CDY	CDY_k	?	CMT_k
EtM	none	CMT_k	CDY	CDY_k	CMT_k	CMT_k
EtM	RBT_k	CMT_k	CDY	CDY_k	CMT_k	CMT_k
KEtM	COLL	SCMT	CDY	CDY_k	CMT	CMT_k
MtE	none	?	?	?	CMT_k	?
MtE	RBT_k	?	?	?	CMT_k	?
SIV	none	?	CDY	CDY_k	?	?
SIV	RBT_k	SCMT	CDY	CDY_k	CMT	?
KSIV	COLL	?	CDY	CDY_k	?	CMT_k

Figure 18: Summary of our Generic Composition Analyses. The entry at row xxx|yyy and column zzz gives the security or attack we establish for the generic AEAD composition xxx with the ivE/key-schedule satisfying yyy and the MAC satisfying zzz. Entries filled in with '?' indicate open problems. Note that all context-discovery reductions require a restriction on the context-selector.

such a KDF may be instantiated with a PRF f as, for example, $\mathsf{KDF}(K) := (f(K,0), f(K,1))$. We now formally state this property.

Collision game COLL **for key schedule.** In the key-schedule-collision game COLL, an adversary $\mathcal{A}$ outputs K_1, K_2; $\mathcal{A}$ wins if

- $K_1 \neq K_2$;
- $K_{m,1} = K_{m,2}$;

where $(K_{e,1}, K_{m,1}) := \mathsf{KS}(K_1)$, $(K_{e,2}, K_{m,2}) := \mathsf{KS}(K_2)$. We write $\mathsf{Adv}^{\mathrm{ksCOLL}}_{\mathsf{KS}}(\mathcal{A})$ to denote the probability that $\mathcal{A}$ wins.

6.2 Summary of Security Analyses

Next, we look at how the various commitment securities of the AEADs constructed using the above composition paradigms depend on the security of the underlying MAC. We consider five security notions for the MAC $(\text{SCMT}, \text{CDY}, \text{CDY}_k, \text{CMT}$ and $\text{CMT}_k)$ which we combine with three possibilities: with an ivE, with a key-robust ivE, or with an ivE and a collision-resistant key-schedule. The results of this analysis are summarized in Fig.18 and the proof of these results is given in App. E.

From these analyses, we can observe that Encrypt-then-MAC with a collision-resistant key-schedule always retains the security of the underlying MAC. We also observe a different behavior between the committing notions and the context-discovery notions. Both context-discovery and key-discovery security are usually passed from the MAC to the AEAD composition, whereas the result is more disparate for the committing notions. We note that to provide context-commitment, the Encrypt-then-MAC paradigm requires a collision-resistant key-schedule, while Encrypt-and-MAC and SIV require only a key-robust ivE.

Finally, we observe that this framework does not apply well to the MAC-then-encrypt constructions—while we found attacks under certain assumptions, we had to leave as open problems several cases where the security implications of the assumptions on the composition were unclear; notably, in stark contrast to the other paradigms, we failed to find a single combination of assumptions on the MAC and the ivE that leads to a provably secure MAC-then-encrypt composition. This is perhaps not surprising, since our analyses in the other paradigms hinge heavily on the property that the MAC output is a part of the final output; as this is not the case in MAC-then-encrypt, it seems rather difficult to relate committing security assumptions on the underlying MAC to that of the AEAD. While there may be contrived joint security assumptions on the ivE and the MAC that lead to some form of committing security in the AEAD, we deem these to be out of scope for this work and leave them as interesting open problems for the future.

Concurrent work. In independent and concurrent work to ours, Struck and Weishäupl [SW24] also analyzed the committing security of some generic composition schemes. Their results are however limited to context-committing security, and to the Encrypt-then-Mac and Encrypt-and-Mac paradigms.

Acknowledgments

We would like to thank Vukašin Karadžić for helpful discussions during the initial stage of this work. We are grateful to the anonymous CRYPTO 2024 reviewers for their constructive comments. Bishwajit Chakraborty was supported by the NRF-ANR project SELECT ("NRF2020-NRF-ANR072"). Wonseok Choi was supported in part by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (No.RS-2023-00239620), by AnalytiXIN, by Sunday Group, and by the Algorand Foundation via the MEGA-ACE project. Jérôme Govinden was supported by the German Federal Ministry of Education and Research and the Hessen State Ministry for Higher Education, Research and the Arts within their joint support of the National Research Center for Applied Cybersecurity ATHENE. Yaobin Shen was supported in parts by Ant Research, Ant Group. This work was partly performed while the authors were attending ASK 2023 at Guangzhou.

References

[ADG+22] Ange Albertini, Thai Duong, Shay Gueron, Stefan Kölbl, Atul Luykx, and Sophie Schmieg. How to abuse and fix authenticated encryption without key commitment. In Kevin R. B. Butler and Kurt Thomas, editors, *USENIX Security 2022: 31st USENIX Security Symposium*, pages 3291–3308, Boston, MA, USA, August 10–12, 2022. USENIX Association. (Cited on pp. 2 and 8.)

[AS11] Elena Andreeva and Martijn Stam. The symbiosis between collision and preimage resistance. In Liqun Chen, editor, *13th IMA International Conference on Cryptography and Coding*, volume 7089 of *Lecture Notes in Computer Science*, pages 152–171, Oxford, UK, December 12–15, 2011. Springer, Heidelberg, Germany. (Cited on p. 10.)

[BBGS23] Matilda Backendal, Mihir Bellare, Felix Günther, and Matteo Scarlata. When messages are keys: Is HMAC a dual-PRF? In Helena Handschuh and Anna Lysyanskaya, editors, *Advances in Cryptology – CRYPTO 2023, Part III*, volume 14083 of *Lecture Notes in Computer Science*, pages 661–693, Santa Barbara, CA, USA, August 20–24, 2023. Springer, Heidelberg, Germany. (Cited on p. 16.)

[BCG+] Mihir Bellare, John Chan, Paul Grubbs, Viet Tung Hoang, Sanketh Menda, Julia Len, Thomas Ristenpart, and Phillip Rogaway. Ask Your Cryptographer if Context-Committing AEAD Is Right for You. Presented at the IACR Real World Crypto Symposium 2023. (Cited on p. 2.)

[BCZ05] Martin Boesgaard, Thomas Christensen, and Erik Zenner. Badger - a fast and provably secure MAC. In John Ioannidis, Angelos Keromytis, and Moti Yung, editors, *ACNS 05: 3rd International Conference on Applied Cryptography and Network Security*, volume 3531 of *Lecture*

Notes in Computer Science, pages 176–191, New York, NY, USA, June 7–10, 2005. Springer, Heidelberg, Germany. (Cited on pp. 16 and 17.)

[BDPV12] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. Duplexing the sponge: Single-pass authenticated encryption and other applications. In Ali Miri and Serge Vaudenay, editors, *SAC 2011: 18th Annual International Workshop on Selected Areas in Cryptography*, volume 7118 of *Lecture Notes in Computer Science*, pages 320–337, Toronto, Ontario, Canada, August 11–12, 2012. Springer, Heidelberg, Germany. (Cited on p. 14.)

[Ber05] Daniel J. Bernstein. The poly1305-AES message-authentication code. In Henri Gilbert and Helena Handschuh, editors, *Fast Software Encryption – FSE 2005*, volume 3557 of *Lecture Notes in Computer Science*, pages 32–49, Paris, France, February 21–23, 2005. Springer, Heidelberg, Germany. (Cited on p. 17.)

[Ber23] Francesco Berti. Reconsidering generic composition: The modes A10, A11 and A12 are insecure. In Leonie Simpson and Mir Ali Rezazadeh Baee, editors, *ACISP 23: 28th Australasian Conference on Information Security and Privacy*, volume 13915 of *Lecture Notes in Computer Science*, pages 157–176, Brisbane, QLD, Australia, July 5–7, 2023. Springer, Heidelberg, Germany. (Cited on p. 24.)

[BH22] Mihir Bellare and Viet Tung Hoang. Efficient schemes for committing authenticated encryption. In Orr Dunkelman and Stefan Dziembowski, editors, *Advances in Cryptology – EUROCRYPT 2022, Part II*, volume 13276 of *Lecture Notes in Computer Science*, pages 845–875, Trondheim, Norway, May 30 – June 3, 2022. Springer, Heidelberg, Germany. (Cited on pp. 2, 4, 9, 10, 11, 12, 13, and 19.)

[BHW] Mihir Bellare, Viet Tung Hoang, and Cong Wu. The Landscape of Committing Authenticated Encryption. Presented at the Third NIST Workshop on Block Cipher Modes of Operation 2023. (Cited on p. 2.)

[BKLW23] Daniel Bourdrez, Dr. Hugo Krawczyk, Kevin Lewi, and Christopher A. Wood. The OPAQUE Asymmetric PAKE Protocol. Internet-Draft draft-irtf-cfrg-opaque-13, Internet Engineering Task Force, December 2023. Work in Progress. (Cited on pp. 7 and 10.)

[BN00] Mihir Bellare and Chanathip Namprempre. Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. In Tatsuaki Okamoto, editor, *Advances in Cryptology – ASIACRYPT 2000*, volume 1976 of *Lecture Notes in Computer Science*, pages 531–545, Kyoto, Japan, December 3–7, 2000. Springer, Heidelberg, Germany. (Cited on p. 23.)

[BPP18] Francesco Berti, Olivier Pereira, and Thomas Peters. Reconsidering generic composition: The tag-then-encrypt case. In Debrup Chakraborty and Tetsu Iwata, editors, *Progress in Cryptology - INDOCRYPT 2018: 19th International Conference in Cryptology in India*, volume 11356 of *Lecture Notes in Computer Science*, pages 70–90, New Delhi, India, December 9–12, 2018. Springer, Heidelberg, Germany. (Cited on p. 24.)

[BPR05] Mihir Bellare, Krzysztof Pietrzak, and Phillip Rogaway. Improved security analyses for CBC MACs. In Victor Shoup, editor, *Advances in Cryptology – CRYPTO 2005*, volume 3621 of *Lecture Notes in Computer Science*, pages 527–545, Santa Barbara, CA, USA, August 14–18, 2005. Springer, Heidelberg, Germany. (Cited on p. 21.)

[BRW04] Mihir Bellare, Phillip Rogaway, and David Wagner. The EAX mode of operation. In Roy and Meier [RM04], pages 389–407. (Cited on p. 4.)

[Che22] Lily Chen. Recommendation for key derivation using pseudorandom functions. NIST Special Publication (SP) 800-108 Rev.1, 2022. (Cited on p. 8.)

[CJN20] Bishwajit Chakraborty, Ashwin Jha, and Mridul Nandi. On the security of sponge-type authenticated encryption modes. *IACR Transactions on Symmetric Cryptology*, 2020(2):93–119, 2020. (Cited on p. 36.)

[CJN21] Soumya Chattopadhyay, Ashwin Jha, and Mridul Nandi. Fine-tuning the ISO/IEC standard LightMAC. In Mehdi Tibouchi and Huaxiong Wang, editors, *Advances in Cryptology – ASIACRYPT 2021, Part III*, volume 13092 of *Lecture Notes in Computer Science*, pages 490–519, Singapore, December 6–10, 2021. Springer, Heidelberg, Germany. (Cited on p. 19.)

[CJN22] Soumya Chattopadhyay, Ashwin Jha, and Mridul Nandi. Towards tight security bounds for OMAC, XCBC and TMAC. In Shweta Agrawal and Dongdai Lin, editors, *Advances in Cryptology – ASIACRYPT 2022, Part I*, volume 13791 of *Lecture Notes in Computer Science*, pages 348–378, Taipei, Taiwan, December 5–9, 2022. Springer, Heidelberg, Germany. (Cited on p. 22.)

[CR22] John Chan and Phillip Rogaway. On committing authenticated-encryption. In Vijayalakshmi Atluri, Roberto Di Pietro, Christian Damsgaard Jensen, and Weizhi Meng, editors, *ESORICS 2022: 27th European Symposium on Research in Computer Security, Part II*, volume 13555 of *Lecture Notes in Computer Science*, pages 275–294, Copenhagen, Denmark, September 26–30, 2022. Springer, Heidelberg, Germany. (Cited on pp. 2 and 4.)

[DFG23] Jean Paul Degabriele, Marc Fischlin, and Jérôme Govinden. The indifferentiability of the duplex and its practical applications. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 237–269. Springer, 2023. (Cited on p. 2.)

[DGGP21] Jean Paul Degabriele, Jérôme Govinden, Felix Günther, and Kenneth G. Paterson. The security of ChaCha20-Poly1305 in the multi-user setting. In Giovanni Vigna and Elaine Shi, editors, *ACM CCS 2021: 28th Conference on Computer and Communications Security*, pages 1981–2003, Virtual Event, Republic of Korea, November 15–19, 2021. ACM Press. (Cited on p. 2.)

[DGRW18] Yevgeniy Dodis, Paul Grubbs, Thomas Ristenpart, and Joanne Woodage. Fast message franking: From invisible salamanders to encryptment. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology – CRYPTO 2018, Part I*, volume 10991 of *Lecture Notes in Computer Science*, pages 155–186, Santa Barbara, CA, USA, August 19–23, 2018. Springer, Heidelberg, Germany. (Cited on p. 2.)

[DH79] Whitfield Diffie and Martin E Hellman. Privacy and authentication: An introduction to cryptography. *Proceedings of the IEEE*, 67(3):397–427, 1979. (Cited on p. 14.)

[DMVA23] Joan Daemen, Silvia Mella, and Gilles Van Assche. Committing authenticated encryption based on shake. *Cryptology ePrint Archive*, 2023. (Cited on p. 2.)

[DRST12] Yevgeniy Dodis, Thomas Ristenpart, John P. Steinberger, and Stefano Tessaro. To hash or not to hash again? (In)differentiability results for H^2 and HMAC. In Safavi-Naini and Canetti [SNC12], pages 348–366. (Cited on pp. 15 and 22.)

[Dwo05] Morris Dworkin. Recommendation for block cipher modes of operation: The cmac mode for authentication. NIST Special Publication (SP) 800-38B, 2005. (Cited on p. 2.)

[Dwo07a] Morris Dworkin. Recommendation for block cipher modes of operation: Galois/counter mode (gcm) and gmac. Technical Report NIST Special Publication (SP) 800-38D, National Institute of Standards and Technology, 2007. (Cited on pp. 2, 4, and 17.)

[Dwo07b] Morris Dworkin. Recommendation for block cipher modes of operation: The ccm mode for authentication and confidentiality. Technical Report NIST Special Publication (SP) 800-38C, National Institute of Standards and Technology, 2007. (Cited on p. 4.)

[EMST78] William F Ehrsam, Carl HW Meyer, John L Smith, and Walter L Tuchman. Message verification and transmission error detection by block chaining, February 14 1978. US Patent 4,074,066. (Cited on p. 14.)

[FOR17] Pooya Farshim, Claudio Orlandi, and Răzvan Roşie. Security of symmetric primitives under incorrect usage of keys. *IACR Transactions on Symmetric Cryptology*, 2017(1):449–473, 2017. (Cited on pp. 2, 3, 4, 8, 10, 14, and 15.)

[FZOeS14] Rui Figueiredo, André Zúquete, and Tomás Oliveira e Silva. Massively parallel identification of privacy-preserving vehicle RFID tags. In *International workshop on radio frequency identification: Security and privacy issues*, pages 36–53. Springer, 2014. (Cited on p. 7.)

[GLR17] Paul Grubbs, Jiahui Lu, and Thomas Ristenpart. Message franking via committing authenticated encryption. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology – CRYPTO 2017, Part III*, volume 10403 of *Lecture Notes in Computer Science*, pages 66–97, Santa Barbara, CA, USA, August 20–24, 2017. Springer, Heidelberg, Germany. (Cited on pp. 2, 3, 4, 8, and 10.)

[Har08] Dan Harkins. Synthetic Initialization Vector (SIV) Authenticated Encryption Using the Advanced Encryption Standard (AES). RFC 5297, October 2008. (Cited on p. 4.)

[HTT18] Viet Tung Hoang, Stefano Tessaro, and Aishwarya Thiruvengadam. The multi-user security of GCM, revisited: Tight bounds for nonce randomization. In David Lie, Mohammad Mannan, Michael Backes, and XiaoFeng Wang, editors, *ACM CCS 2018: 25th Conference on Computer and Communications Security*, pages 1429–1440, Toronto, ON, Canada, October 15–19, 2018. ACM Press. (Cited on p. 2.)

[IEE19] IEEE. IEEE Standard for Authenticated Encryption with Length Expansion for Storage Devices. IEEE Std 1619.1-2018 (Revision of IEEE Std 1619.1-2007), 2019. (Cited on p. 4.)

[IK03] Tetsu Iwata and Kaoru Kurosawa. OMAC: One-key CBC MAC. In Thomas Johansson, editor, *Fast Software Encryption – FSE 2003*, volume 2887 of *Lecture Notes in Computer Science*, pages 129–153, Lund, Sweden, February 24–26, 2003. Springer, Heidelberg, Germany. (Cited on p. 22.)

[IOM12] Tetsu Iwata, Keisuke Ohashi, and Kazuhiko Minematsu. Breaking and repairing GCM security proofs. In Safavi-Naini and Canetti [SNC12], pages 31–49. (Cited on p. 18.)

[ISO99] ISO. ISO/IEC:Information technology – Security techniques – Entity authentication – Part 4: Mechanisms using a cryptographic check function. Iso/iec 9798-4:1999, International Organization for Standardization, 1999. (Cited on p. 7.)

[ISO11a] ISO. ISO/IEC:Information technology – Security techniques – Message Authentication Codes (MACs) – Part 1: Mechanisms using a block cipher. Iso/iec 9797-1:2011, International Organization for Standardization, 2011. (Cited on pp. 4, 14, 15, and 19.)

[ISO11b] ISO. ISO/IEC:Information technology – Security techniques – Message Authentication Codes (MACs) – Part 3: Mechanisms using a universal hash-function. Iso/iec 9797-3:2011, International Organization for Standardization, 2011. (Cited on pp. 4, 14, and 16.)

[ISO17] ISO. ISO:Banking and related financial services – Key wrap using AES. Iso 20038:2017, International Organization for Standardization, 2017. (Cited on p. 4.)

[ISO19] ISO. ISO/IEC:Information technology – Lightweight cryptography – Part 6: Message authentication codes (MACs). Iso/iec 29192-6:2019, International Organization for Standardization, 2019. (Cited on pp. 4, 14, and 18.)

[ISO20] ISO. Information security – authenticated encryption. Standard ISO/IEC 19772:2020, International Organization for Standardization, Geneva, CH, 2020. (Cited on p. 4.)

[ISO21] ISO. ISO/IEC:Information technology – Security techniques – Message Authentication Codes (MACs) – Part 2: Mechanisms using a dedicated hash-function. Iso/iec 9797-1:2021, International Organization for Standardization, 2021. (Cited on pp. 4, 14, 15, 16, and 17.)

[ISO23] ISO. ISO/IEC:Information technology – Automatic identification and data capture techniques – Part 11: Crypto suite PRESENT-80 security services for air interface communications. Iso/iec 29167-11:2023, International Organization for Standardization, 2023. (Cited on p. 7.)

[JKX18] Stanislaw Jarecki, Hugo Krawczyk, and Jiayu Xu. OPAQUE: An asymmetric PAKE protocol secure against pre-computation attacks. In Jesper Buus Nielsen and Vincent Rijmen, editors, *Advances in Cryptology – EUROCRYPT 2018, Part III*, volume 10822 of *Lecture Notes in Computer Science*, pages 456–486, Tel Aviv, Israel, April 29 – May 3, 2018. Springer, Heidelberg, Germany. (Cited on p. 7.)

[JN16] Ashwin Jha and Mridul Nandi. Revisiting structure graph and its applications to CBC-MAC and EMAC. *IACR Cryptol. ePrint Arch.*, page 161, 2016. (Cited on p. 21.)

[KBC97] Dr. Hugo Krawczyk, Mihir Bellare, and Ran Canetti. HMAC: Keyed-Hashing for Message Authentication. RFC 2104, February 1997. (Cited on p. 2.)

[KE10] Dr. Hugo Krawczyk and Pasi Eronen. HMAC-based Extract-and-Expand Key Derivation Function (HKDF). RFC 5869, May 2010. (Cited on p. 8.)

[KR14] Ted Krovetz and Phillip Rogaway. The OCB Authenticated-Encryption Algorithm. RFC 7253, May 2014. (Cited on p. 4.)

[KSW23] Juliane Krämer, Patrick Struck, and Maximiliane Weishäupl. Committing authenticated encryption: Sponges vs. block-ciphers in the case of the nist lwc finalists. *Cryptology ePrint Archive*, 2023. (Cited on p. 2.)

[LGR20] Julia Len, Paul Grubbs, and Thomas Ristenpart. Partitioning oracle attacks. Cryptology ePrint Archive, Report 2020/1491, 2020. `https://eprint.iacr.org/2020/1491`. (Cited on p. 17.)

[LGR21] Julia Len, Paul Grubbs, and Thomas Ristenpart. Partitioning oracle attacks. In Michael Bailey and Rachel Greenstadt, editors, *USENIX Security 2021: 30th USENIX Security Symposium*, pages 195–212. USENIX Association, August 11–13, 2021. (Cited on p. 2.)

[LGR22] Julia Len, Paul Grubbs, and Thomas Ristenpart. Authenticated encryption with key identification. In Shweta Agrawal and Dongdai Lin, editors, *Advances in Cryptology – ASIACRYPT 2022, Part III*, volume 13793 of *Lecture Notes in Computer Science*, pages 181–209, Taipei, Taiwan, December 5–9, 2022. Springer, Heidelberg, Germany. (Cited on p. 7.)

[LPTY16] Atul Luykx, Bart Preneel, Elmar Tischhauser, and Kan Yasuda. A MAC mode for lightweight block ciphers. In Thomas Peyrin, editor, *Fast Software Encryption – FSE 2016*, volume 9783 of *Lecture Notes in Computer Science*, pages 43–59, Bochum, Germany, March 20–23, 2016. Springer, Heidelberg, Germany. (Cited on pp. 18 and 19.)

[MLGR23] Sanketh Menda, Julia Len, Paul Grubbs, and Thomas Ristenpart. Context discovery and commitment attacks - how to break CCM, EAX, SIV, and more. In Carmit Hazay and Martijn Stam, editors, *Advances in Cryptology – EUROCRYPT 2023, Part IV*, volume 14007 of *Lecture Notes in Computer Science*, pages 379–407, Lyon, France, April 23–27, 2023. Springer, Heidelberg, Germany. (Cited on pp. 2, 9, 11, 12, and 23.)

[MMV$^+$14] Nicky Mouha, Bart Mennink, Anthony Van Herrewege, Dai Watanabe, Bart Preneel, and Ingrid Verbauwhede. Chaskey: An efficient MAC algorithm for 32-bit microcontrollers. In Antoine Joux and Amr M. Youssef, editors, *SAC 2014: 21st Annual International Workshop on Selected Areas in Cryptography*, volume 8781 of *Lecture Notes in Computer Science*, pages 306–323, Montreal, QC, Canada, August 14–15, 2014. Springer, Heidelberg, Germany. (Cited on p. 19.)

[MST] John Preuß Mattsson, Ben Smeets, and Erik Thormarker. Proposals for standardization of encryption schemes. Presented at the Third NIST Workshop on Block Cipher Modes of Operation 2023. (Cited on p. 2.)

[MSW06] David Molnar, Andrea Soppera, and David Wagner. A scalable, delegatable pseudonym protocol enabling ownership transfer of RFID tags. In Bart Preneel and Stafford Tavares, editors, *SAC 2005: 12th Annual International Workshop on Selected Areas in Cryptography*, volume 3897 of *Lecture Notes in Computer Science*, pages 276–290, Kingston, Ontario, Canada, August 11–12, 2006. Springer, Heidelberg, Germany. (Cited on p. 7.)

[MV04] David A. McGrew and John Viega. The security and performance of the Galois/counter mode (GCM) of operation. In Anne Canteaut and Kapalee Viswanathan, editors, *Progress in Cryptology - INDOCRYPT 2004: 5th International Conference in Cryptology in India*, volume 3348 of *Lecture Notes in Computer Science*, pages 343–355, Chennai, India, December 20–22, 2004. Springer, Heidelberg, Germany. (Cited on pp. 17 and 18.)

[MW04] David Molnar and David Wagner. Privacy and security in library RFID: Issues, practices, and architectures. In Vijayalakshmi Atluri, Birgit Pfitzmann, and Patrick McDaniel, editors, *ACM CCS 2004: 11th Conference on Computer and Communications Security*, pages 210–219, Washington, DC, USA, October 25–29, 2004. ACM Press. (Cited on p. 7.)

[Nan09] Mridul Nandi. Improved security analysis for OMAC as a pseudorandom function. *J. Math. Cryptol.*, 3(2):133–148, 2009. (Cited on p. 22.)

[Nan18] Mridul Nandi. Bernstein bound on WCS is tight - repairing luykx-preneel optimal forgeries. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology – CRYPTO 2018, Part II*, volume 10992 of *Lecture Notes in Computer Science*, pages 213–238, Santa Barbara, CA, USA, August 19–23, 2018. Springer, Heidelberg, Germany. (Cited on p. 17.)

[NL18] Yoav Nir and Adam Langley. ChaCha20 and Poly1305 for IETF Protocols. RFC 8439, June 2018. (Cited on pp. 2 and 4.)

[NRS14] Chanathip Namprempre, Phillip Rogaway, and Thomas Shrimpton. Reconsidering generic composition. In Phong Q. Nguyen and Elisabeth Oswald, editors, *Advances in Cryptology – EUROCRYPT 2014*, volume 8441 of *Lecture Notes in Computer Science*, pages 257–274, Copenhagen, Denmark, May 11–15, 2014. Springer, Heidelberg, Germany. (Cited on pp. 3, 4, 23, 24, and 25.)

[NSS23] Yusuke Naito, Yu Sasaki, and Takeshi Sugawara. Committing security of ascon: Cryptanalysis on primitive and proof on mode. *IACR Transactions on Symmetric Cryptology*, 2023(4):420–451, 2023. (Cited on p. 2.)

[PCS$^+$05] Adrian Perrig, Ran Canetti, Dawn Song, Professor Doug Tygar, and Bob Briscoe. Timed Efficient Stream Loss-Tolerant Authentication (TESLA): Multicast Source Authentication Transform Introduction. RFC 4082, June 2005. (Cited on p. 8.)

[PCTS00] Adrian Perrig, Ran Canetti, J. Doug Tygar, and Dawn Xiaodong Song. Efficient authentication and signing of multicast streams over lossy channels. In *2000 IEEE Symposium on Security and Privacy*, pages 56–73, Oakland, CA, USA, May 2000. IEEE Computer Society Press. (Cited on p. 8.)

[Pv95] Bart Preneel and Paul C. van Oorschot. MDx-MAC and building fast MACs from hash functions. In Don Coppersmith, editor, *Advances in Cryptology – CRYPTO'95*, volume 963 of *Lecture Notes in Computer Science*, pages 1–14, Santa Barbara, CA, USA, August 27–31, 1995. Springer, Heidelberg, Germany. (Cited on p. 15.)

[RM04] Bimal K. Roy and Willi Meier, editors. *Fast Software Encryption – FSE 2004*, volume 3017 of *Lecture Notes in Computer Science*, New Delhi, India, February 5–7, 2004. Springer, Heidelberg, Germany. (Cited on pp. 28 and 33.)

[Rog04] Phillip Rogaway. Efficient instantiations of tweakable blockciphers and refinements to modes OCB and PMAC. In Pil Joong Lee, editor, *Advances in Cryptology – ASIACRYPT 2004*, volume 3329 of *Lecture Notes in Computer Science*, pages 16–31, Jeju Island, Korea, December 5–9, 2004. Springer, Heidelberg, Germany. (Cited on p. 19.)

[RS04] Phillip Rogaway and Thomas Shrimpton. Cryptographic hash-function basics: Definitions, implications, and separations for preimage resistance, second-preimage resistance, and collision resistance. In Roy and Meier [RM04], pages 371–388. (Cited on pp. 6, 10, and 20.)

[Sho96] Victor Shoup. On fast and provably secure message authentication based on universal hashing. In Neal Koblitz, editor, *Advances in Cryptology – CRYPTO'96*, volume 1109 of *Lecture Notes in Computer Science*, pages 313–328, Santa Barbara, CA, USA, August 18–22, 1996. Springer, Heidelberg, Germany. (Cited on p. 17.)

[SNC12] Reihaneh Safavi-Naini and Ran Canetti, editors. *Advances in Cryptology – CRYPTO 2012*, volume 7417 of *Lecture Notes in Computer Science*, Santa Barbara, CA, USA, August 19–23, 2012. Springer, Heidelberg, Germany. (Cited on pp. 29 and 30.)

[Stä22] Mirco Stäubl. Actually good encryption? confusing users by changing nonces. Semester project, Department of Computer Science, ETH Zürich, July 2022. (Cited on pp. 8 and 15.)

[SW24] Patrick Struck and Maximiliane Weishäupl. Constructing committing and leakage-resilient authenticated encryption. *IACR Transactions on Symmetric Cryptology*, 2024(1):497–528, Mar. 2024. (Cited on p. 27.)

[TSL08] Chiu C Tan, Bo Sheng, and Qun Li. Secure and serverless RFID authentication and search protocols. *IEEE Transactions on Wireless Communications*, 7(4):1400–1407, 2008. (Cited on p. 7.)

[WC81] Mark N. Wegman and Larry Carter. New Hash Functions and Their Use in Authentication and Set Equality. *J. Comput. Syst. Sci.*, 22(3):265–279, 1981. (Cited on p. 16.)

[WSRE04] Stephen A Weis, Sanjay E Sarma, Ronald L Rivest, and Daniel W Engels. Security and privacy aspects of low-cost radio frequency identification systems. In *Security in Pervasive Computing: First International Conference, Boppard, Germany, March 12-14, 2003. Revised Papers*, pages 201–212. Springer, 2004. (Cited on p. 7.)

A Relations between Security Notions

In this section, we prove the relations between AEAD or MAC security notions as represented in Fig. 3 and recalled in the following proposition.

Proposition 2. *Given any* AE *or* MAC *scheme, the following security relations hold:*

1. *it is* SCMT *secure if and only if it is both* CDY *and* CMT *secure,*
2. *if it is* CMT *secure then it is* CMT_k *secure,*
3. *if it is* CMT_k *secure then it is* RBT_k *secure,*
4. *if it is* CDY *secure then it is* CDY_k *secure for a family of context-selectors,*
5. *if it is* CMT *secure then it is* CDY_k *secure for a family of context-selectors.*

Proof. Let $\mathsf{MAC} = (\mathsf{MAC.Tg}, \mathsf{MAC.Vf})$ be a message authentication code. Relations (1), (2), and (3) follow directly from the definitions. In the following, we prove the relations for MACs, but the proof is similar for AEADs.

We now prove relation (4). Let S be a restricted context selector that can be modeled as a sequence of algorithms $(\mathsf{S}_1, \mathsf{S}_2)$, where S runs both S_1 and S_2, S_1 outputs (N, M), S_2 outputs T and S combines both results to output (N, M, T). Let $\mathcal{A}$ be a CDY_k adversary, we define an adversary $\mathcal{B}_{\mathsf{S}_1}$ against the CDY game as a procedure that runs S_1 and obtains a pair (N, M) it combines it with its CDY challenge T and runs $\mathcal{A}(N, M, T)$ to obtain a key K. As output to the CDY game, $\mathcal{B}_{\mathsf{S}_1}$ returns (K, N, M). By construction, $\mathcal{B}_{\mathsf{S}_1}$ when run with the context-selector S_2 as CDY challenge, simulate exactly the game CDY_k played by $\mathcal{A}$ with the context-selector S as challenge. Thus $\mathsf{Adv}^{\mathrm{macCDY_k}}_{\mathsf{MAC}}(\mathcal{A}; \mathsf{S}) = \mathsf{Adv}^{\mathrm{macCDY}}_{\mathsf{MAC}}(\mathcal{B}_{\mathsf{S}_1}; \mathsf{S}_2)$.

We now prove relation (5). For any (K, N, M), let S be a restricted context selector for the CDY_k game that returns a tuple (N', M', T), such that $(N', M') \neq (N, M)$ and $T = \mathsf{MAC.Tg}(K, N, M)$. Let $\mathcal{A}$ be a CDY_k adversary, we define an adversary $\mathcal{B}_{\mathsf{S}}$ against the CDY game as a procedure that runs S and obtains a tuple (N', M', T) it then runs $\mathcal{A}(N', M', T)$ to obtain a key K'. As output to the CMT game, $\mathcal{B}_{\mathsf{S}}$ returns $(K, N, M), (K', N', M')$. As (N', M', T) was obtained through S, we have that always $(N', M') \neq (N, M)$ and $T = \mathsf{MAC.Tg}(K, N, M)$. By construction, when $\mathcal{B}_{\mathsf{S}}$ is run it simulate exactly the game CDY_k played by $\mathcal{A}$ with the context-selector S as challenge. Thus $\mathsf{Adv}^{\mathrm{macCDY_k}}_{\mathsf{MAC}}(\mathcal{A}; \mathsf{S}) = \mathsf{Adv}^{\mathrm{macCMT}}_{\mathsf{MAC}}(\mathcal{B}_{\mathsf{S}})$.

□

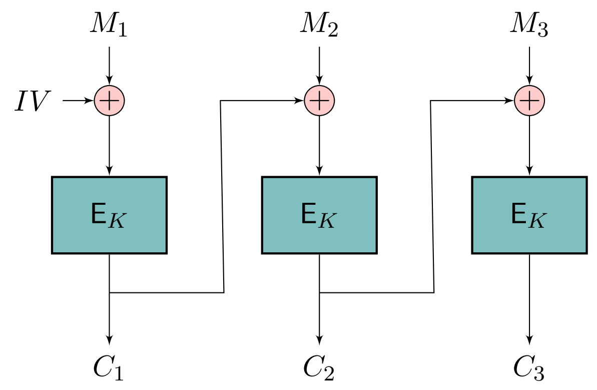

Figure 19: CBC Encryption.

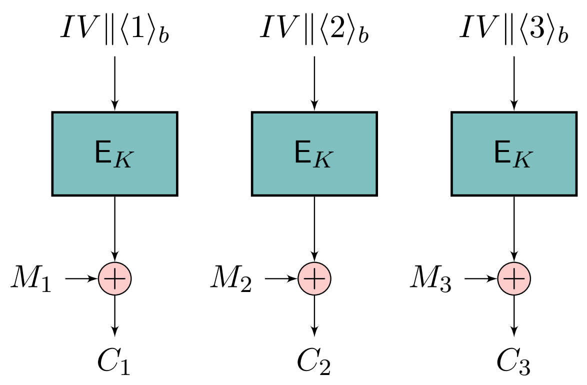

Figure 20: CTR Encryption with block cipher of state size n. $b := n - |IV|$.

B Key-Robustness Security Analysis of CBC/CTR in the Ideal-Cipher Model

Proposition 3. *For any adversary $\mathcal{A}$ that tries to break the key robustness of the* $\mathsf{CBC/CTR}$ *encryption scheme by making at most σ calls to an ideal cipher* E *of state size n,*

$$\mathsf{Adv}^{\mathrm{seRBT_k}}_{\mathsf{CBC[E]/CTR[E]}}(\mathcal{A}, \sigma) \leq \frac{\sigma^2}{2^n}.$$

Proof. For any such adversary $\mathcal{A}$, consider another key robustness adversary $\mathcal{B}$ against the same ideal cipher E which makes at most σ queries.

Note that $\mathcal{B}$ can act as a key-robustness challenger to $\mathcal{A}$ as follows:

- Whenever $\mathcal{A}$ makes a forward/backward ideal cipher query of the form (K, X) to E, $\mathcal{B}$ makes the same ideal cipher query and sends the received response to $\mathcal{A}$.
- Whenever, $\mathcal{A}$ produces (K_1, K_2, IV, M) as the challenge output, then $\mathcal{B}$ produces $((K_1, K_2), X)$ as its challenge output where

$$X := \begin{cases} IV \oplus \lfloor M \rfloor_n & \text{for } \mathsf{CBC} \\ IV\|\langle 1\rangle_{n-|IV|} & \text{for } \mathsf{CTR}. \end{cases}$$

From the construction of $\mathsf{CBC/CTR}$, it easily follows that if $\mathcal{A}$ wins in the key-robustness game against $\mathsf{CBC[E]/CTR[E]}$ then the adversary $\mathcal{B}$ wins the key-robustness game against the ideal cipher E.

Finally, the proposition follows from the observation that in the ideal cipher model

$$\mathsf{Adv}^{\mathrm{seRBT_k}}_{\mathsf{E}}(\mathcal{B}, \sigma) \leq \frac{\sigma^2}{2^n}.$$

□

C Key-Robustness Security Analysis of the Duplex in the Ideal Permutation Model

Consider an IV-based Duplex encryption scheme where κ, c, b respectively denote the key size, capacity size, and state size of the Duplex. For simplicity, we assume IV is of size $b - \kappa$. For the exact construction

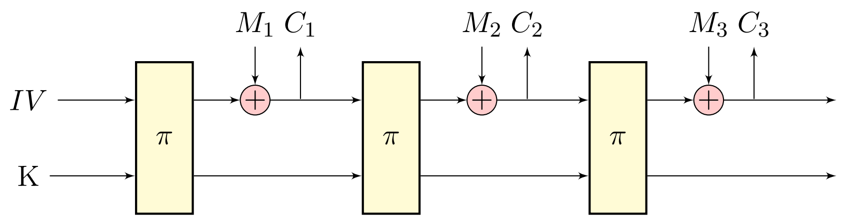

Figure 21: Duplex Encryption.

of the scheme, we refer to Fig. 21.

Proposition 4. *For any adversary $\mathcal{A}$ that tries to break the key robustness of the* Duplex *by making at most σ calls to an ideal permutation of state size b,*

$$\mathsf{Adv}^{\mathrm{seRBT_k}}_{\mathsf{Duplex}}(\mathcal{A},\sigma) \leq \frac{\sigma^2}{2^{b-\max\{\kappa,c\}}}.$$

Proof. For any such adversary $\mathcal{A}$, we consider another adversary $\mathcal{B}$ with oracle access to the same ideal permutation π and makes at most σ queries. We say that $\mathcal{B}$ wins if it can come up with (U_1, U_2) such that $U_1 \neq U_2$, $\lfloor U_1 \rfloor_{b-\kappa} = \lfloor U_2 \rfloor_{b-\kappa}$ and $\lfloor \pi(U_1) \rfloor_{b-c} = \lfloor \pi(U_2) \rfloor_{b-c}$.

Note that $\mathcal{B}$ can act as a key-robustness challenger to $\mathcal{A}$ as follows:

- Whenever $\mathcal{A}$ makes a forward/backward ideal permutation query of the form X, $\mathcal{B}$ makes the same ideal permutation query and sends the received response to $\mathcal{A}$.
- Whenever, $\mathcal{A}$ produces (K_1, K_2, IV, M) as the challenge output, then B produces $(IV\|K_1, IV\|K_2)$ as it's challenge output.

Claim 1. *If $\mathcal{A}$ wins in the key-robustness game then the adversary $\mathcal{B}$ wins.*

Proof. Let $C := \mathsf{SpongeEnc}(K_1, IV, M)$ then if $\mathcal{A}$ wins we must have $\mathsf{SpongeEnc}(K_2, IV, M) = C$ and hence $\lfloor \pi(IV\|K_1) \rfloor_{b-c} = \lfloor \pi(IV\|K_2) \rfloor_{b-c} = \lfloor M \oplus C \rfloor_{b-c}$. □

Lemma 2. *If the adversary $\mathcal{B}$ in Proposition 4 can make at most σ forward/backward queries to the random permutation π then in the ideal permutation model,*

$$Pr[\mathcal{B}\ wins] \leq \frac{\sigma^2}{2^{b-\max\{\kappa,c\}}}.$$

Proof. Let $\{(U_i, V_i, dir_i) | i \in [1, \sigma]\}$ denote the permutation query transcript for $\mathcal{B}$ where $dir_i = +$ if the ith query is a forward query and $-$ otherwise. Then $\mathcal{B}$ wins only if one of the following three events occurs.

- $\exists i \neq j \in [1, \sigma]$ s.t. $dir_i = dir_j = +$ and $\lfloor V_i \rfloor_{b-c} = \lfloor V_j \rfloor_{b-c}$. In the ideal permutation model probability that this happens is bounded by $\frac{\sigma^2}{2^{b-c}}$.
- $\exists i \neq j \in [1, \sigma]$ s.t. $dir_i = dir_j = -$ and $\lfloor U_i \rfloor_{b-\kappa} = \lfloor U_j \rfloor_{b-\kappa}$. In the ideal permutation model probability that this happens is bounded by $\frac{\sigma^2}{2^{b-\kappa}}$.
- $\exists i \neq j \in [1, \sigma]$ s.t. $dir_i = +, dir_j = -$ and $\lfloor U_i \rfloor_{b-\kappa} = \lfloor U_j \rfloor_{b-\kappa}$ and $\lfloor V_i \rfloor_{b-c} = \lfloor V_j \rfloor_{b-c}$. Following the definition of expected maximum multicollision in truncated random sampling without replacement in [CJN20], the probability of this case can be bounded by $\frac{\sigma^2}{2^{2b-c-\kappa}}$.

Combining all three cases we have lemma 2. □

Finally, Proposition 4 follows from Claim 1 and Lemma 2.

□

D Davies-Meyer Collision Resistance and Preimage Resistance Proof

D.1 Davies-Meyer collision resistance proof

Let (K, X, Y) be the entry that records the query and the response of E and E^{-1}, where K is the key, X the plaintext and Y the ciphertext. Let C_i be the event that a collision is found for DM at the i-th ideal-cipher query. It implies that for some $j < i$,

$$E_{K_i}(X_i) \oplus X_i = E_{K_j}(X_j) \oplus X_j$$

which happens with probability at most $(i-1)/(2^n - i + 1)$ as $E_{K_i}(X_i) \oplus X_i$ is uniformly distributed in a set of size at least $2^n - i + 1$ regardless of it comes from the forward or the backward query to E. Hence,

$$\mathsf{Adv}^{\mathrm{coll}}_{\mathsf{DM}}(\mathcal{A}) \le \sum_{i=1}^{p} \Pr[C_j] \le \sum_{i=1}^{p} \frac{i-1}{2^n - i + 1} \le \frac{p(p-1)}{2(2^n - p + 1)} \le \frac{p^2}{2^n}$$

by assuming $p \le 2^{n-1}$.

D.2 Davies-Meyer preimage resistance proof

Let (K, X, Y) be entry that records the query and the response of E and E^{-1}, where K is the key, X the plaintext and Y the ciphertext Y. Fix a image $T \in \{0,1\}^n$. Let P_i be the event that the preimage of T is found for DM at the i-th ideal-cipher query. Then, it implies

$$E_{K_i}(X_i) \oplus X_i = T$$

if it is a query to E, or

$$E^{-1}_{K_i}(Y_i) \oplus Y_i = T$$

if it is a query to E^{-1}. In either of these two cases, the equation happens with probability at most $1/(2^n - i + 1)$ as both $E_{K_i}(X_i) \oplus X_i$ and $E^{-1}_{K_i}(Y_i) \oplus Y_i$ are uniformly distributed in a set of size at least $2^n - i + 1$. Hence,

$$\mathsf{Adv}^{\mathrm{epre}}_{\mathsf{DM}}(\mathcal{A}) \le \sum_{i=1}^{p} \Pr[P_i] \le \frac{2p}{2^n}$$

by assuming $p \le 2^{n-1}$.

E Security Analyses of Generic Composition

In this section, we look at how the various commitment securities of the AEADs constructed using the composition paradigms described in Sec. 6.1 depend on the commitment security of the underlying MAC and the robustness of the underlying ivE.

We note that for all of our strong-committing, context-discovery, or key-discovery reductions, we prove the result only for a restricted form of context-selector (which is a sequence of algorithms similar to the proof of Sec. A). For simplicity and wlog, we also assume a bijective padding pad for Encrypt-then-MAC and SIV.

E.1 Encrypt-and-MAC.

We start by analyzing Encrypt-and-MAC where for the two variants, the encryption functions are described in Fig. 13 and the decryption functions in Fig. 22.

```
$\mathsf{AE}_1^{\mathsf{EaM}}[\mathsf{SE},\mathsf{MAC}].\mathsf{Dec}(K,N,A,C,T)$
1 : $IV \leftarrow N$
2 : $(K_e, K_m) \leftarrow K$
3 : $M \leftarrow \mathsf{SE.Dec}(K_e, IV, C)$
4 : $M^* \leftarrow \mathsf{pad}(A, M)$
5 : if $\mathsf{MAC.Vf}(K_m, N, M^*, T) = 1$
6 :     return $M$
7 : else
8 :     return $\perp$
9 : endif

$\mathsf{AE}_2^{\mathsf{EaM}}[\mathsf{SE},\mathsf{MAC}].\mathsf{Dec}(K,N,A,C,T)$
1 : $A^* \leftarrow \mathsf{pad}(A)$
2 : $IV \leftarrow \mathsf{MAC.Tg}(K_m, N, A^*)$
3 : $(K_e, K_m) \leftarrow K$
4 : $M \leftarrow \mathsf{SE.Dec}(K_e, IV, C)$
5 : $M^* \leftarrow \mathsf{pad}(A, M)$
6 : if $\mathsf{MAC.Vf}(K_m, N, M^*, T) = 1$
7 :     return $M$
8 : else
9 :     return $\perp$
10 : endif
```

Figure 22: Encrypt-and-MAC decryption: Variant 1 (left) and Variant 2 (right).

Strong-Committing Security. In the Encrypt-and-MAC composition paradigm, we will show that the strong-commitment security carries over from the MAC to the AEAD, as long as the ivE is robust. More specifically, for any AEAD-strong-committing adversary $\mathcal{A}$ and a restricted context-selector S for $\mathsf{AE}^{\mathsf{EaM}}[\mathsf{SE}, \mathsf{MAC}]$, we can construct an ivE-key-robustness adversary $\mathcal{A}_e$ and an MAC-key-discovery adversary $\mathcal{A}_m$ which simulate $\mathcal{A}$, and show that there is a context selector $\widetilde{\mathsf{S}}$ for MAC such that if $\mathcal{A}$ wins against $(\mathsf{AE}^{\mathsf{EaM}}[\mathsf{SE},\mathsf{MAC}], \mathsf{S})$, then $\mathcal{A}_e$ wins against SE or $\mathcal{A}_m$ wins against $(\mathsf{MAC}, \widetilde{\mathsf{S}})$. Precisely, we have the following lemma.

Lemma 3. *For any AEAD-strong-committing adversary $\mathcal{A}$ and a restricted context-selector $\mathsf{S} = (\mathsf{S}_1, \mathsf{S}_2)$ for $\mathsf{AE}^{\mathsf{EaM}}[\mathsf{SE},\mathsf{MAC}]$, there exist an ivE-key-robustness adversary $\mathcal{A}_e$ and a MAC-strong-committing adversary $\mathcal{A}_m$ such that*

$$\mathsf{Adv}^{\mathrm{aeadSCMT}}_{\mathsf{AE}^{\mathsf{EaM}}[\mathsf{SE},\mathsf{MAC}]}(\mathcal{A};\mathsf{S}) \leq \mathsf{Adv}^{\mathrm{seRBT_k}}_{\mathsf{SE}}(\mathcal{A}_e) + \mathsf{Adv}^{\mathrm{macSCMT}}_{\mathsf{MAC}}(\mathcal{A}_m^{\mathsf{S}_2};\mathsf{S}_1).$$

where S runs both S_1 and S_2, S_1 outputs T, S_2 outputs C, and S combines both results to output (C, T).

Proof. We need to show that if $\mathcal{A}$ wins against $(\mathsf{AE}^{\mathsf{EaM}}[\mathsf{SE},\mathsf{MAC}], \mathsf{S})$, then either $\mathcal{A}_e$ wins against SE or $\mathcal{A}_m$ wins against $(\mathsf{MAC}, \widetilde{\mathsf{S}})$. Suppose, on receiving the challenge $C^\dagger := (C, T)$, $\mathcal{A}$ outputs $(K_1 = (K_{e,1}, K_{m,1}), N_1, A_1, M_1)$ and $(K_2 = (K_{e,2}, K_{m,2}), N_2, A_2, M_2)$; then $\mathcal{A}_e$ outputs $(K_{e,1}, K_{e,2}, N_1, M_1)$, and $\mathcal{A}_m$ outputs $(K_{m,1}, N_1, M_1^* := \mathsf{pad}(A_1, M_1))$ and $(K_{m,2}, N_2, M_2^* := \mathsf{pad}(A_2, M_2))$. We will show this for $\mathsf{AE}_1^{\mathsf{EaM}}[\mathsf{SE},\mathsf{MAC}]$ and omit a proof for $\mathsf{AE}_2^{\mathsf{EaM}}[\mathsf{SE},\mathsf{MAC}]$ since it is almost identical. Suppose $\mathcal{A}$ wins.

1. If $\mathsf{AE}_1^{\mathsf{EaM}}[\mathsf{SE},\mathsf{MAC}].\mathsf{Enc}(K_1, N_1, A_1, M_1) = (C, T)$ (i.e., $\mathcal{A}$ won by context discovery), then T must be $\mathsf{MAC.Tg}(K_{m,1}, N_1, M_1^*)$, so $\mathcal{A}_m$ wins by context discovery in the following way: $\mathcal{A}_m$ receives its challenge from S_1 and internally runs S_2 so it simulates exactly $\mathsf{S} = (\mathsf{S}_1, \mathsf{S}_2)$ for $\mathcal{A}$.

2. Otherwise, we must have $(K_1, N_1, A_1, M_1) \neq (K_2, N_2, A_2, M_2)$, and $\mathsf{AE}_1^{\mathsf{EaM}}[\mathsf{SE},\mathsf{MAC}].\mathsf{Enc}(K_1, N_1, A_1, M_1) = \mathsf{AE}_1^{\mathsf{EaM}}[\mathsf{SE},\mathsf{MAC}].\mathsf{Enc}(K_2, N_2, A_2, M_2)$ (i.e., $\mathcal{A}$ won by breaking context commitment). The second condition implies that $\mathsf{SE.Enc}(K_{e,1}, N_1, M_1) = \mathsf{SE.Enc}(K_{e,2}, N_2, M_2)$ and $\mathsf{MAC.Tg}(K_{m,1}, N_1, M_1^*) = \mathsf{MAC.Tg}(K_{m,2}, N_2, M_2^*)$. Now, we have two possibilities:

   - if $(K_{m,1}, N_1, M_1^*) \neq (K_{m,2}, N_2, M_2^*)$, $\mathcal{A}_m$ also wins by breaking context-commitment for MAC;
   - if $(K_{m,1}, N_1, M_1^*) = (K_{m,2}, N_2, M_2^*)$, since pad is injective, we must have $K_{e,1} \neq K_{e,2}$, in which case $\mathcal{A}_e$ wins by breaking key-robustness for SE.

This completes the proof. $\square$

It may be interesting to note here that the bound on the context-commitment security of $\mathsf{AE}^{\mathsf{EaM}}[\mathsf{SE}, \mathsf{MAC}]$ is strict—a key-robustness attack on SE or a context-committing attack on MAC can be translated into a context-committing attack on $\mathsf{AE}^{\mathsf{EaM}}[\mathsf{SE}, \mathsf{MAC}]$. However, the same is not true for the context-discovery security of $\mathsf{AE}^{\mathsf{EaM}}[\mathsf{SE}, \mathsf{MAC}]$, since a context-discovery attack on MAC is (in general) not sufficient to mount a context-discovery attack on $\mathsf{AE}^{\mathsf{EaM}}[\mathsf{SE}, \mathsf{MAC}]$.

We note that a similar analysis follows for the reduction of the CMT security of EaM to the security of the MAC.

Key-committing security with key-scheduling. For key-schedule-based Encrypt-and-MAC, a similar analysis holds for key-committing security as follows.

Lemma 4. *For any AEAD-key-committing adversary $\mathcal{A}$, there exist a key-schedule-collision adversary $\mathcal{A}_k$ and a MAC-key-committing adversary $\mathcal{A}_m$ such that*

$$\mathsf{Adv}^{\mathrm{aeadCMT_k}}_{\mathsf{kAE}^{\mathsf{EaM}}[\mathsf{SE},\mathsf{MAC}]}(\mathcal{A}) \le \mathsf{Adv}^{\mathrm{ksCOLL}}_{\mathsf{KS}}(\mathcal{A}_k) + \mathsf{Adv}^{\mathrm{macCMT_k}}_{\mathsf{MAC}}(\mathcal{A}_m).$$

Proof. We show this for $\mathsf{kAE}_1^{\mathsf{EaM}}[\mathsf{KS}, \mathsf{SE}, \mathsf{MAC}]$. The proof for $\mathsf{kAE}_2^{\mathsf{EaM}}[\mathsf{KS}, \mathsf{SE}, \mathsf{MAC}]$ is almost identical, so we skip it. Similarly from the proof of Lemma 4, we argue that if $\mathcal{A}$ wins against ($\mathsf{kAE}^{\mathsf{EaM}}[\mathsf{KS}, \mathsf{SE}, \mathsf{MAC}]$), then either $\mathcal{A}_k$ wins against KS or $\mathcal{A}_m$ wins against MAC. Suppose $\mathcal{A}$ outputs (K_1, N_1, A_1, M_1) and (K_2, N_2, A_2, M_2), such that $\mathsf{KS}(K_1) = (K_{e,1}, K_{m,1})$, $\mathsf{KS}(K_2) = (K_{e,2}, K_{m,2})$; then $\mathcal{A}_k$ outputs (K_1, K_2), and $\mathcal{A}_m$ outputs $(K_{m,1}, N_1, M_1^* := \mathsf{pad}(A_1, M_1))$ and $(K_{m,2}, N_2, M_2^* := \mathsf{pad}(A_2, M_2))$. Suppose $\mathcal{A}$ wins. Then we must have $K_1 \neq K_2$, and $\mathsf{kAE}_1^{\mathsf{EaM}}[\mathsf{KS}, \mathsf{SE}, \mathsf{MAC}].\mathsf{Enc}(K_1, N_1, A_1, M_1) = \mathsf{kAE}_1^{\mathsf{EaM}}[\mathsf{KS}, \mathsf{SE}, \mathsf{MAC}].\mathsf{Enc}(K_2, N_2, A_2, M_2)$. The second condition implies that $\mathsf{SE}.\mathsf{Enc}(K_{e,1}, N_1, M_1) = \mathsf{SE}.\mathsf{Enc}(K_{e,2}, N_2, M_2)$, and $\mathsf{MAC}.\mathsf{Tg}(K_{m,1}, N_1, M_1^*) = \mathsf{MAC}.\mathsf{Tg}(K_{m,2}, N_2, M_2^*)$. Now, we have two possibilities:

- if $K_{m,1} \neq K_{m,2}$, $\mathcal{A}_m$ also wins by breaking key-commitment for MAC;
- if $K_{m,1} = K_{m,2}$, $\mathcal{A}_k$ wins by finding a collision for KS.

This completes the proof. $\square$

Key-discovery security. In the Encrypt-and-MAC composition paradigm, we will show that the key-discovery security carries over from the MAC to the AEAD.

Lemma 5. *For any AEAD-key-discovery adversary $\mathcal{A}$ and a restricted context-selector $\mathsf{S} = (\mathsf{S}_1, \mathsf{S}_2)$ for $\mathsf{AE}^{\mathsf{EaM}}[\mathsf{SE}, \mathsf{MAC}]$, there exist an MAC-key-discovery adversary $\mathcal{B}^{\mathsf{S}_2}$ and a context selector $\widetilde{\mathsf{S}}$ for MAC such that,*

$$\mathsf{Adv}^{\mathrm{aeadCDY_k}}_{\mathsf{AE}^{\mathsf{EaM}}[\mathsf{SE},\mathsf{MAC}]}(\mathcal{A}; \mathsf{S}) \le \mathsf{Adv}^{\mathrm{macCDY_k}}_{\mathsf{MAC}}(\mathcal{B}^{\mathsf{S}_2}; \widetilde{\mathsf{S}})$$

where S runs both S_1 and S_2, S_1 outputs (N, A, M, T), S_2 outputs C and S combines both results to output $(N, A, M, C^\dagger := (C, T))$.

Proof. Let $\widetilde{\mathsf{S}}$ be the context selector that runs S_1, and when S_1 outputs (N, A, M, T), $\widetilde{\mathsf{S}}$ outputs the challenge $(N, M^* := \mathsf{pad}(A, M), T)$. Let $\mathcal{A}$ be a CDY$_k$ adversary for EaM, we define an adversary $\mathcal{B}^{\mathsf{S}_2}$ against the MAC CDY$_k$ game as a procedure that runs S_2 and obtains C it combines it with its MAC CDY$_k$ challenge $(N, M^* := \mathsf{pad}(A, M), T)$ and runs $\mathcal{A}(N, A, M, (C, T))$ to obtain a key $K = (K_e, K_m)$. As output to the MAC CDY$_k$ game, $\mathcal{B}^{\mathsf{S}_2}$ returns K_m.

By construction, $\mathcal{B}^{\mathsf{S}_2}$ when run with the context-selector $\widetilde{\mathsf{S}}$ as MAC CDY$_k$ challenge, simulate exactly the game CDY$_k$ played by $\mathcal{A}$ with the context-selector S as challenge.

$\square$

A similar analysis follows for the reduction of the CDY security of EaM to the security of the MAC.

```
AE_1^EtM[SE, MAC].Dec(K, N, A, C, T)
1 :  $IV \leftarrow N$
2 :  $(K_e, K_m) \leftarrow K$
3 :  $C^* \leftarrow \mathsf{pad}(A, C)$
4 :  if $\mathsf{MAC.Vf}(K_m, N, C^*, T) = 1$
5 :    $M \leftarrow \mathsf{SE.Dec}(K_e, IV, C)$
6 :    return $M$
7 :  else
8 :    return $\perp$
9 :  endif
```

```
AE_2^EtM[SE, MAC].Dec(K, N, A, C, T)
1 :  $A^* \leftarrow \mathsf{pad}(A)$
2 :  $IV \leftarrow \mathsf{MAC.Tg}(K_m, N, A^*)$
3 :  $(K_e, K_m) \leftarrow K$
4 :  $C^* \leftarrow \mathsf{pad}(A, C)$
5 :  if $\mathsf{MAC.Vf}(K_m, N, C^*, T) = 1$
6 :    $M \leftarrow \mathsf{SE.Dec}(K_e, IV, C)$
7 :    return $M$
8 :  else
9 :    return $\perp$
10 : endif
```

Figure 23: Encrypt-then-MAC decryption: Variant 1 (left) and Variant 2 (right).

E.2 Encrypt-then-MAC.

For Encrypt-then-MAC, the encryption functions are described in Fig. 14 and the decryption functions in Fig. 23.

In the Encrypt-then-MAC composition paradigm, irrespective of the security of SE and MAC, we can construct a constant-query adversary $\mathcal{A}$ which wins with probability 1 in an AEAD-key-committing game against $\mathsf{AE}^{\mathsf{EtM}}[\mathsf{SE}, \mathsf{MAC}]$. We show this for $\mathsf{AE}_1^{\mathsf{EtM}}[\mathsf{SE}, \mathsf{MAC}]$; an almost identical attacks works for $\mathsf{AE}_2^{\mathsf{EaM}}[\mathsf{SE}, \mathsf{MAC}]$, so we skip it.

1. $\mathcal{A}$ picks any $K_1 = (K_{e,1}, K_m)$, N, A, and M_1 and computes $(C, T) = \mathsf{AE}_1^{\mathsf{EtM}}[\mathsf{SE}, \mathsf{MAC}].\mathsf{Enc}(K_1, N, A, M_1)$;
2. $\mathcal{A}$ picks any $K_{e,2} \neq K_{e,1}$, sets $K_2 := (K_{e,2}, K_m)$, and computes $M_2 = \mathsf{AE}_1^{\mathsf{EtM}}[\mathsf{SE}, \mathsf{MAC}].\mathsf{Dec}(K_2, N, A, C, T)$;
3. $\mathcal{A}$ outputs (K_1, N, A, M_1) and (K_2, N, A, M_2).

In Step 2, $\mathcal{A}$ will not receive $\perp$ by the correctness of MAC, and $\mathsf{AE}_1^{\mathsf{EtM}}[\mathsf{SE}, \mathsf{MAC}].\mathsf{Enc}(K_2, N, A, M_2) = (C, T)$ by the correctness of $\mathsf{AE}_1^{\mathsf{EtM}}[\mathsf{SE}, \mathsf{MAC}]$. Thus, $\mathcal{A}$ always wins.

Key-schedule-based Encrypt-then-MAC. In the key-schedule-based variants of Encrypt-then-MAC, the above attack does not work if KS is secure against collision attacks. Most parts of the proofs are similar to those of the previous Encrypt-and-MAC results.

Strong-committing Security. We will show that $\mathsf{kAE}^{\mathsf{EtM}}[\mathsf{KS}, \mathsf{SE}, \mathsf{MAC}]$ is strong-committing if KS is collision-resistant and MAC is strong-committing. $\widetilde{\mathsf{S}}$ runs S, and when S outputs a challenge $C^\dagger := (C, T)$, $\widetilde{\mathsf{S}}$ outputs the challenge T. Suppose $\mathcal{A}$ outputs (K_1, N_1, A_1, M_1) and (K_2, N_2, A_2, M_2), such that $\mathsf{KS}(K_1) = (K_{e,1}, K_{m,1})$, and $\mathsf{KS}(K_2) = (K_{e,2}, K_{m,2})$; then $\mathcal{A}_k$ outputs (K_1, K_2), and $\mathcal{A}_m$ outputs $(K_{m,1}, N_1, C_1^* := \mathsf{pad}(A_1, C)$ and $(K_{m,2}, N_2, C_2^* := \mathsf{pad}(A_2, C))$. Then we can show the following result.

Lemma 6. *For any AEAD-strong-committing adversary $\mathcal{A}$ and a restricted context-selector $\mathsf{S} = (\mathsf{S}_1, \mathsf{S}_2)$ for $\mathsf{kAE}^{\mathsf{EtM}}[\mathsf{KS}, \mathsf{SE}, \mathsf{MAC}]$, there exist a key-schedule-collision adversary $\mathcal{A}_k$ and a MAC-strong-committing adversary $\mathcal{A}_m$ such that*

$$\mathsf{Adv}^{\mathrm{aeadSCMT}}_{\mathsf{kAE}^{\mathsf{EtM}}[\mathsf{KS},\mathsf{SE},\mathsf{MAC}]}(\mathcal{A}; \mathsf{S}) \leq \mathsf{Adv}^{\mathrm{ksCOLL}}_{\mathsf{KS}}(\mathcal{A}_k) + \mathsf{Adv}^{\mathrm{macSCMT}}_{\mathsf{MAC}}(\mathcal{A}_m^{\mathsf{S}_1}; \mathsf{S}_2)$$

where S runs both S_1 and S_2, S_1 outputs T, S_2 outputs C, and S combines both results to output (C, T).

Proof. We will see if $\mathcal{A}$ wins against $(\mathsf{kAE}^{\mathsf{EtM}}[\mathsf{KS}, \mathsf{SE}, \mathsf{MAC}], \mathsf{S})$, then either $\mathcal{A}_e$ wins against SE or $\mathcal{A}_m$ wins against $(\mathsf{MAC}, \widetilde{\mathsf{S}})$. We show this for $\mathsf{kAE}_1^{\mathsf{EtM}}[\mathsf{KS}, \mathsf{SE}, \mathsf{MAC}]$. The proof for $\mathsf{kAE}_2^{\mathsf{EtM}}[\mathsf{KS}, \mathsf{SE}, \mathsf{MAC}]$ is almost identical, so we skip it. Suppose $\mathcal{A}$ wins.

1. If $\mathsf{kAE}_1^{\mathsf{EtM}}[\mathsf{KS}, \mathsf{SE}, \mathsf{MAC}].\mathsf{Enc}(K_1, N_1, A_1, M_1) = (C, T)$ (i.e., $\mathcal{A}$ won by context discovery), then T must be $\mathsf{MAC}.\mathsf{Tg}(K_{m,1}, N_1, C_1^*)$, so $\mathcal{A}_m$ can simulate $\mathcal{A}$ and also wins by context discovery.

2. Otherwise, we must have $(K_1, N_1, A_1, M_1) \neq (K_2, N_2, A_2, M_2)$, and $\mathsf{kAE}_1^{\mathsf{EtM}}[\mathsf{KS}, \mathsf{SE}, \mathsf{MAC}].\mathsf{Enc}(K_1, N_1, A_1, M_1) = \mathsf{kAE}_1^{\mathsf{EtM}}[\mathsf{KS}, \mathsf{SE}, \mathsf{MAC}].\mathsf{Enc}(K_2, N_2, A_2, M_2)$. The second condition implies that $\mathsf{SE}.\mathsf{Enc}(K_{e,1}, N_1, M_1) = \mathsf{SE}.\mathsf{Enc}(K_{e,2}, N_2, M_2)$, and $\mathsf{MAC}.\mathsf{Tg}(K_{m,1}, N_1, C_1^*) = \mathsf{MAC}.\mathsf{Tg}(K_{m,2}, N_2, C_2^*)$. Now, we have two possibilities:

   - if $(K_{m,1}, N_1, A_1) \neq (K_{m,2}, N_2, A_2)$, $\mathcal{A}_m$ wins by breaking context-commitment for MAC;
   - if $(K_{m,1}, N_1, A_1) = (K_{m,2}, N_2, A_2)$, we must have $(K_1, M_1) \neq (K_2, M_2)$; now, having $K_1 = K_2$ would force $K_{e,1} = K_{e,2}$, which in turn would force $M_1 = M_2$ (since SE is an encryption scheme), contradicting the requirement $(K_1, M_1) \neq (K_2, M_2)$; so we must have $K_1 \neq K_2$, in which case $\mathcal{A}_k$ wins by finding a collision for KS.

This completes the proof. □

We note that a similar analysis follows for the reduction of the CMT security of KEtM to the security of the MAC.

Key-committing security with key-scheduling. A similar analysis holds for key-committing security.

Lemma 7. *For any AEAD-key-committing adversary $\mathcal{A}$, we can construct a key-schedule-collision adversary $\mathcal{A}_k$ and a MAC-key-committing adversary $\mathcal{A}_m$*

$$\mathsf{Adv}^{\mathrm{aeadCMT_k}}_{\mathsf{kAE}^{\mathsf{EtM}}[\mathsf{SE},\mathsf{MAC}]}(\mathcal{A}) \leq \mathsf{Adv}^{\mathrm{ksCOLL}}_{\mathsf{KS}}(\mathcal{A}_k) + \mathsf{Adv}^{\mathrm{macCMT_k}}_{\mathsf{MAC}}(\mathcal{A}_m).$$

Proof. We show this for $\mathsf{kAE}_1^{\mathsf{EtM}}[\mathsf{KS}, \mathsf{SE}, \mathsf{MAC}]$. Suppose $\mathcal{A}$ outputs (K_1, N_1, A_1, M_1) and (K_2, N_2, A_2, M_2), such that $\mathsf{KS}(K_1) = (K_{e,1}, K_{m,1})$, $\mathsf{KS}(K_2) = (K_{e,2}, K_{m,2})$; then $\mathcal{A}_k$ outputs (K_1, K_2), and $\mathcal{A}_m$ outputs $(K_{m,1}, N_1, C_1^* := \mathsf{pad}(A_1, C))$ and $(K_{m,2}, N_2, C_2^* := \mathsf{pad}(A_2, C))$. Suppose $\mathcal{A}$ wins. Then we must have $K_1 \neq K_2$, and $\mathsf{kAE}_1^{\mathsf{EtM}}[\mathsf{KS}, \mathsf{SE}, \mathsf{MAC}].\mathsf{Enc}(K_1, N_1, A_1, M_1) = \mathsf{kAE}_1^{\mathsf{EtM}}[\mathsf{KS}, \mathsf{SE}, \mathsf{MAC}].\mathsf{Enc}(K_2, N_2, A_2, M_2)$. The second condition implies that $\mathsf{SE}.\mathsf{Enc}(K_{e,1}, N_1, M_1) = \mathsf{SE}.\mathsf{Enc}(K_{e,2}, N_2, M_2)$, and $\mathsf{MAC}.\mathsf{Tg}(K_{m,1}, N_1, C_1^*) = \mathsf{MAC}.\mathsf{Tg}(K_{m,2}, N_2, C_2^*)$. Now, we have two possibilities:

- if $K_{m,1} \neq K_{m,2}$, $\mathcal{A}_m$ also wins by breaking key-commitment for MAC;
- if $K_{m,1} = K_{m,2}$, $\mathcal{A}_k$ wins by finding a collision for KS.

This completes the proof. □

```
$\mathsf{AE}_1^{\mathsf{MtE}}[\mathsf{SE}, \mathsf{MAC}].\mathsf{Dec}(K, N, A, C^\dagger)$
$A^* \leftarrow \mathsf{pad}(A)$
$IV \leftarrow \mathsf{MAC.Tg}(K_m, N, A^*)$
$(K_e, K_m) \leftarrow K$
$(M, T) \leftarrow \mathsf{SE.Dec}(K_e, IV, C^\dagger)$
$M^* \leftarrow \mathsf{pad}(A, M)$
if $\mathsf{MAC.Vf}(K_m, N, M^*, T) = 1$
    return $M$
else
    return $\perp$
endif
```

```
$\mathsf{AE}_2^{\mathsf{MtE}}[\mathsf{SE}, \mathsf{MAC}].\mathsf{Dec}(K, N, A, C^\dagger)$
$IV \leftarrow N$
$(K_e, K_m) \leftarrow K$
$(M, T) \leftarrow \mathsf{SE.Dec}(K_e, IV, C^\dagger)$
$M^* \leftarrow \mathsf{pad}(A, M)$
if $\mathsf{MAC.Vf}(K_m, N, M^*, T) = 1$
    return $M$
else
    return $\perp$
endif
```

Figure 24: MAC-then-Encrypt decryption: Variant 1 (left) and Variant 2 (right).

Key-discovery security. In the Encrypt-then-MAC design paradigm, like in Encrypt-and-MAC, we can show that key-discovery security carries over from the MAC to the AEAD, even in the absence of a key schedule.

Lemma 8. *For any AEAD-key-discovery adversary $\mathcal{A}$ and a restricted context-selector $\mathsf{S} = (\mathsf{S}_1, \mathsf{S}_2)$ for $\mathsf{AE}^{\mathsf{EaM}}[\mathsf{SE}, \mathsf{MAC}]$, there exist an MAC-key-discovery adversary $\mathcal{B}^{\mathsf{S}_2}$ and a context selector $\widetilde{\mathsf{S}}$ for MAC such that,*

$$\mathsf{Adv}^{\mathrm{aeadCDY_k}}_{\mathsf{AE}^{\mathsf{EtM}}[\mathsf{SE},\mathsf{MAC}]}(\mathcal{A}; \mathsf{S}) \leq \mathsf{Adv}^{\mathrm{macCDY_k}}_{\mathsf{MAC}}(\mathcal{B}^{\mathsf{S}_2}; \widetilde{\mathsf{S}})$$

where S runs both S_1 and S_2, S_1 outputs (N, A, C, T), S_2 outputs M and S combines both results to output $(N, A, M, C^\dagger := (C, T))$.

Proof. Let $\widetilde{\mathsf{S}}$ be the context selector that runs S_1, and when S_1 outputs (N, A, C, T), $\widetilde{\mathsf{S}}$ outputs the challenge $(N, M^* := \mathsf{pad}(A, C), T)$. Let $\mathcal{A}$ be a CDY$_k$ adversary for EtM, we define an adversary $\mathcal{B}^{\mathsf{S}_2}$ against the MAC CDY$_k$ game as a procedure that runs S_2 and obtains M it combines it with its MAC CDY$_k$ challenge $(N, M^* := \mathsf{pad}(A, C), T)$ and runs $\mathcal{A}(N, A, M, (C, T))$ to obtain a key $K = (K_e, K_m)$. As output to the MAC CDY$_k$ game, $\mathcal{B}^{\mathsf{S}_2}$ returns K_m.

By construction, $\mathcal{B}^{\mathsf{S}_2}$ when run with the context-selector $\widetilde{\mathsf{S}}$ as MAC CDY$_k$ challenge, simulate exactly the game CDY$_k$ played by $\mathcal{A}$ with the context-selector S as challenge. □

A similar analysis follows for the reduction of the CDY security of EaM to the security of the MAC.

E.3 MAC-then-Encrypt.

For MAC-then-Encrypt, the encryption functions are described in Fig. 15 and the decryption functions in Fig. 24. We exhibit the following attack.

We can prove that for any encryption scheme SE, there exists a context-committing MAC for which the composition is insecure. This notably proves that key robust encryption and context-committing MAC are not sufficient conditions to obtain key commitment of MAC-then-Encrypt.

1. Assume there exists a context committing MAC called MAC. Define n, a, $k_m \neq k'_m$ and $k_e \neq k'_e$.

2. Let $t = \mathsf{MAC.Tg}(k_m, n, a, \varepsilon)$, $c = \mathsf{SE.Enc}(k_e, n, t)$ and $t' = \mathsf{SE.Dec}(k'_e, n, c)$ (i.e., $c = \mathsf{SE.Enc}(k'_e, n, t')$).

3. If $t = t'$, then $((k_e, k'_e), (k_m, k_m), (n, n), (a, a), (\varepsilon, \varepsilon), c)$ breaks the key committing security of MAC-then-Enc.

4. If $t \neq t'$, define

$$\mathsf{MAC.Tg}'(K_m, N, A, M) = \begin{cases} t' & \textbf{if } (K_m, N, A, M) = (k'_m, n, a, \varepsilon) \\ \mathsf{MAC.Tg}(k'_m, n, a, \varepsilon) & \textbf{elseif } \mathsf{MAC.Tg}(K_m, N, A, M) = t' \\ \mathsf{MAC.Tg}(K_m, N, A, M) & \text{otherwise} \end{cases}$$

   Then $((k_e, k'_e), (k_m, k'_m), (n, n), (a, a), (\varepsilon, \varepsilon), c)$ breaks the key committing security of $\mathsf{MAC.Tg}'$-then-Enc. Note that $\mathsf{MAC.Tg}'$ is context committing if MAC is context committing.

E.4 SIV.

The SIV encryption and decryption functions are described in Fig. 16.

Strong-committing security. In the SIV composition paradigm, like in Encrypt-and-MAC, we will show that the commitment security carries over from the MAC to the AEAD, as long as the ivE is robust.

Lemma 9. *For any AEAD-strong-committing adversary $\mathcal{A}$ and a restricted context-selector $\mathsf{S} = (\mathsf{S}_1, \mathsf{S}_2)$ for $\mathsf{AE}^{\mathsf{SIV}}[\mathsf{SE}, \mathsf{MAC}]$, there exist an ivE-key-robustness adversary $\mathcal{A}_e$ and a MAC-strong-committing adversary $\mathcal{A}_m$ such that*

$$\mathsf{Adv}^{\text{aeadSCMT}}_{\mathsf{AE}^{\mathsf{SIV}}[\mathsf{SE},\mathsf{MAC}]}(\mathcal{A}; \mathsf{S}) \leq \mathsf{Adv}^{\text{seRBT}_\text{k}}_{\mathsf{SE}}(\mathcal{A}_e) + \mathsf{Adv}^{\text{macSCMT}}_{\mathsf{MAC}}(\mathcal{A}_m^{\mathsf{S}_2}; \mathsf{S}_1).$$

where S runs both S_1 and S_2, S_1 outputs T, S_2 outputs C, and S combines both results to output (C, T).

Proof. Similarly to the previous proofs, on receiving the challenge (C, T), $\mathcal{A}$ outputs $(K_1 = (K_{e,1}, K_{m,1}), N_1, A_1, M_1)$ and $(K_2 = (K_{e,2}, K_{m,2}), N_2, A_2, M_2)$; then $\mathcal{A}_e$ outputs $(K_{e,1}, K_{e,2}, N_1, M_1)$, and $\mathcal{A}_m$ outputs $(K_{m,1}, N_1, M_1^* := \mathsf{pad}(A_1, M_1))$ and $(K_{m,2}, N_2, M_2^* := \mathsf{pad}(A_2, M_2))$.

If $\mathsf{AE}^{\mathsf{SIV}}[\mathsf{SE}, \mathsf{MAC}].\mathsf{Enc}(K_1, N_1, A_1, M_1) = (C, T)$ (i.e., $\mathcal{A}$ won by context discovery), then T must be $\mathsf{MAC.Tg}(K_{m,1}, N_1, M_1^*)$, so $\mathcal{A}_m$ also wins by context discovery by simulating $\mathcal{A}$. Otherwise, we must have $(K_1, N_1, A_1, M_1) \neq (K_2, N_2, A_2, M_2)$, and $\mathsf{AE}^{\mathsf{SIV}}[\mathsf{SE}, \mathsf{MAC}].\mathsf{Enc}(K_1, N_1, A_1, M_1) = \mathsf{AE}^{\mathsf{SIV}}[\mathsf{SE}, \mathsf{MAC}].\mathsf{Enc}(K_2, N_2, A_2, M_2)$ (i.e., $\mathcal{A}$ won by breaking context commitment).

The second condition implies that $\mathsf{SE.Enc}(K_{e,1}, T_1, M_1) = \mathsf{SE.Enc}(K_{e,1}, T_2, M_2)$, and $\mathsf{MAC.Tg}(K_{m,1}, N_1, M_1^*) = \mathsf{MAC.Tg}(K_{m,2}, N_2, M_2^*)$. Now, we have two possibilities:

- if $(K_{m,1}, N_1, M_1^*) \neq (K_{m,2}, N_2, M_2^*)$, $\mathcal{A}_m$ also wins by breaking context-commitment for MAC;
- if $(K_{m,1}, N_1, M_1^*) = (K_{m,2}, N_2, M_2^*)$, since pad is injective, we must have $K_{e,1} \neq K_{e,2}$, in which case $\mathcal{A}_e$ wins by breaking key-robustness for SE.

This completes the proof. □

As in the case of Encrypt-and-MAC, the bound on the context-commitment security of $\mathsf{AE}^{\mathsf{SIV}}[\mathsf{SE}, \mathsf{MAC}]$ is strict—a key-robustness attack on SE or a context-committing attack on MAC can be translated into a context-committing attack on $\mathsf{AE}^{\mathsf{SIV}}[\mathsf{SE}, \mathsf{MAC}]$. However, the same is not true for the context-discovery security of $\mathsf{AE}^{\mathsf{SIV}}[\mathsf{SE}, \mathsf{MAC}]$, since a context-discovery attack on MAC is (in general) not sufficient to mount a context-discovery attack on $\mathsf{AE}^{\mathsf{SIV}}[\mathsf{SE}, \mathsf{MAC}]$.

We note that a similar analysis follows for the reduction of the CMT security of SIV to the security of the MAC.

Key-committing security with key-scheduling. For key-schedule-based SIV, a similar analysis holds for key-committing security.

Lemma 10. *For any AEAD-key-committing adversary $\mathcal{A}$, there exist a key-schedule-collision adversary $\mathcal{A}_k$ and a MAC-key-committing adversary $\mathcal{A}_m$ such that*

$$\mathsf{Adv}^{\mathrm{aeadCMT_k}}_{\mathsf{kAE^{SIV}[SE,MAC]}}(\mathcal{A}) \leq \mathsf{Adv}^{\mathrm{ksCOLL}}_{\mathsf{KS}}(\mathcal{A}_k) + \mathsf{Adv}^{\mathrm{macCMT_k}}_{\mathsf{MAC}}(\mathcal{A}_m).$$

Proof. Suppose $\mathcal{A}$ outputs (K_1, N_1, A_1, M_1) and (K_2, N_2, A_2, M_2), such that $\mathsf{KS}(K_1) = (K_{e,1}, K_{m,1})$, $\mathsf{KS}(K_2) = (K_{e,2}, K_{m,2})$; then $\mathcal{A}_k$ outputs (K_1, K_2), and $\mathcal{A}_m$ outputs $(K_{m,1}, N_1, M_1^* := \mathsf{pad}(A_1, M_1))$ and $(K_{m,2}, N_2, M_2^* := \mathsf{pad}(A_2, M_2))$.

Now, suppose $\mathcal{A}$ wins. Then we must have $K_1 \neq K_2$, and $\mathsf{kAE^{SIV}[KS, SE, MAC].Enc}(K_1, N_1, A_1, M_1) = \mathsf{kAE^{SIV}[KS, SE, MAC].Enc}(K_2, N_2, A_2, M_2)$. The second condition implies that $\mathsf{SE.Enc}(K_{e,1}, T_1, M_1) = \mathsf{SE.Enc}(K_{e,1}, T_2, M_2)$, and $\mathsf{MAC.Tg}(K_{m,1}, N_1, M_1^*) = \mathsf{MAC.Tg}(K_{m,2}, N_2, M_2^*)$. Now, we have two possibilities:

- if $K_{m,1} \neq K_{m,2}$, $\mathcal{A}_m$ also wins by breaking key-commitment for MAC;
- if $K_{m,1} = K_{m,2}$, $\mathcal{A}_k$ wins by finding a collision for KS.

This completes the proof. $\square$

Key-discovery security. In the SIV paradigm, like in Encrypt-and-MAC, we can show that the key-discovery security carries over from the MAC to the AEAD.

Lemma 11. *For any AEAD-key-discovery adversary $\mathcal{A}$ and a restricted context-selector $\mathsf{S} = (\mathsf{S}_1, \mathsf{S}_2)$ for $\mathsf{AE^{SIV}[SE, MAC]}$, there exist an MAC-key-discovery adversary $\mathcal{A}$ and a context selector $\widetilde{\mathsf{S}}$ for MAC such that,*

$$\mathsf{Adv}^{\mathrm{aeadCDY_k}}_{\mathsf{AE^{SIV}[SE,MAC]}}(\mathcal{A}; \mathsf{S}) \leq \mathsf{Adv}^{\mathrm{macCDY_k}}_{\mathsf{MAC}}(\mathcal{A}_m; \widetilde{\mathsf{S}}).$$

where S runs both S_1 and S_2, S_1 outputs (N, A, M, T), S_2 outputs C and S combines both results to output $(N, A, M, C^\dagger := (C, T))$.

Proof. We let $\widetilde{\mathsf{S}}$ run S, and when S outputs a challenge $(N, A, M, C^\dagger := (C, T))$, $\widetilde{\mathsf{S}}$ outputs the challenge $(N, M^* := \mathsf{pad}(A, M), T)$. Suppose, on receiving the challenge $(N, A, M, C^\dagger)$, $\mathcal{A}$ outputs $K = (K_e, K_m)$; then $\mathcal{A}_m$ outputs K_m. Suppose $\mathcal{A}$ wins, $\mathsf{AE^{SIV}[SE, MAC]} = (K, N, A, M) = C^\dagger$. This in turn implies that $\mathsf{MAC.Tg}(K_m, N, M) = T$, so $\mathcal{A}_m$ also wins by key-discovery. Thus the bound follows. $\square$