

# Committing Security of Forkcipher-based AEAD

Samir Kundu<sup>1</sup>  , Sougata Mandal<sup>2,3</sup>   and Hrithik Nandi<sup>2,4</sup>  

<sup>1</sup> Siksha ‘O’ Anusandhan (Deemed to be) University, Bhubaneswar, India

<sup>2</sup> Institute for Advancing Intelligence (IAI), TCG CREST, Kolkata, India

<sup>3</sup> Umeå University, Umeå, Sweden

<sup>4</sup> Ramakrishna Mission Vivekananda Educational and Research Institute, Belur, India

**Abstract.** Authenticated Encryption with Associated Data (AEAD) provides confidentiality and integrity, but traditional security notions do not prevent ciphertexts from decrypting correctly under multiple keys. This motivates committing security, which requires ciphertexts to bind to their encryption inputs uniquely. Recent analyses revealed commitment failures in several forkcipher-based AEAD schemes, leaving open the question of whether these failures are inherent to forkciphers or design specific. We provide a comprehensive committing security analysis of existing forkcipher-based AEAD constructions, filling this gap by establishing both negative and positive results. Our analysis shows that while some designs inherently fail to achieve strong commitment, others achieve meaningful commitment guarantees, demonstrating that forkciphers can serve as a foundation for committing-secure AEAD schemes. We derive tight bounds for commitment notions for existing constructions. As a key technical contribution, we formalize a forkcipher-based hash construction and prove its collision resistance, which serves as a foundation for our security analysis.

**Keywords:** Committing security · AEAD · RPAEF · FEDT · ForkDTE

## 1 Introduction

Authenticated Encryption with Associated Data (AEAD) is the standard primitive for achieving confidentiality and integrity in symmetric-key cryptography. Under classical definitions, an AEAD scheme ensures that ciphertexts reveal no information about the plaintext and that any modification of the ciphertext or associated data is detected during decryption. However, these guarantees do not preclude the existence of ciphertexts that decrypt correctly under multiple keys. Such behaviour, although outside the scope of conventional AEAD security, has been shown to lead to practical vulnerabilities, including the Facebook message-franking attack [DGRW18] and the partitioning-oracle attack [LGR21], where ambiguity enables key-substitution, cross-context replay, and abuse-report forgery.

To address this limitation, stronger notions such as *robustness* and *committing security* have been introduced. Robustness, initially studied for public-key encryption and later extended to symmetric primitives [Rog04, ABN18], requires that a ciphertext should not verify under multiple keys. Building on this, committing security requires that a ciphertext uniquely binds to its encryption inputs. Informally, no efficient adversary should be able to produce two distinct encryption contexts that result in the same ciphertext, thereby eliminating ambiguity. Commitment notions form a hierarchy, ranging from the weakest notion of key commitment (formalized as CMT1) to the strongest notion of context commitment (formalized as CMT4), with intermediate notions CMT2 and CMT3 capturing commitment to progressively larger subsets of the encryption inputs. However, for any

---

E-mail: [samirkundu3@gmail.com](mailto:samirkundu3@gmail.com) (Samir Kundu), [sougatamandal2014@gmail.com](mailto:sougatamandal2014@gmail.com) (Sougata Mandal), [hrithiknandi.crypto@gmail.com](mailto:hrithiknandi.crypto@gmail.com) (Hrithik Nandi)

correct AEAD scheme, the CMT3 and CMT4 notions are equivalent. Specifically, CMT1 captures key commitment, where the ciphertext is bound only to the key; CMT2 extends this to binding of the ciphertext to the key and nonce; CMT3 further requires binding of the ciphertext to the key, nonce, and associated data; and CMT4 captures full context commitment, requiring the ciphertext to be bound to all encryption inputs.

Over time, these notions have evolved from feasibility results to refined security frameworks. Early works proposed generic transforms achieving partial forms of commitment [FOR17, GLR17, DGRW18], but did not fully capture commitment to all inputs, particularly the key and nonce. Subsequent attacks demonstrated practical weaknesses arising from this gap [LGR21, ADG<sup>+</sup>22], motivating stronger formulations. More recent works formalize hierarchies of commit-to-input notions and unified frameworks for reasoning about committing security [BH22, CR22, MLGR23], along with constructions achieving these guarantees. These developments have also influenced standardization efforts, where commitment is now recognized as a desirable AEAD property [Boz25].

Forkciphers were first introduced by Andreeva *et al.* [ALP<sup>+</sup>19], and subsequent works [KLL20, Man24] explored provably secure forkcipher designs. [ALP<sup>+</sup>19] also proposed several authenticated encryption schemes based on the ForkSkinny primitive, including PAEF, RPAEF, and SAEF, primarily targeting efficient encryption of short messages in resource-constrained environments. Subsequent works further analyzed these schemes and established additional security properties [ABV20, BAV24].

More recently, forkciphers have been used in the design of the Eevee family of AEAD schemes: Umbreon, Jolteon, and Espeon proposed in [BPA<sup>+</sup>23], targeting high-performance and parallelizable encryption in cloud and IoT settings. Forkcipher-based designs have also been employed in leakage-resistant authenticated encryption schemes such as FEDT, FEDT\* [DDL24], ForkDTE1, and ForkDTE2 [BSL24], aimed at secure computation settings that resist side-channel leakage.

Thus, the landscape of forkcipher-based authenticated encryption includes PAEF, RPAEF, SAEF, the Eevee family: Umbreon, Jolteon, and Espeon, as well as leakage-resistant schemes such as FEDT, FEDT\*, ForkDTE1, and ForkDTE2. Given their increasing use in performance-critical and distributed environments, it is important to understand their security under stronger notions such as committing security, particularly since such guarantees are required in modern applications. In this direction, Rahman *et al.* [RKI24] showed that PAEF, SAEF, and the Eevee family fail to achieve the weakest notion of key-commitment security.<sup>1</sup> However, they could not resolve the cases of RPAEF, FEDT, FEDT\*, ForkDTE1, and ForkDTE2. This leaves open the question of whether the failure of committing security is inherent to the forkcipher paradigm or specific to particular constructions.

## 1.1 Contributions

We address the above question by investigating whether committing security can be achieved for the remaining forkcipher-based AEAD schemes. In particular, we study the tweakable forkcipher-based constructions RPAEF, FEDT, and FEDT\* as well as the forkcipher-based schemes ForkDTE1 and ForkDTE2, none of which have been analyzed from the perspective of committing security in prior work.

We begin in Section 3 by analyzing the committing security of RPAEF. We show that RPAEF is insecure under both the CMT1 and CMT2 notions when the adversary is allowed to output empty messages. If the adversary is restricted to outputting only non-empty messages, then RPAEF achieves tight  $n/2$ -bit CMT2 security. We further demonstrate that it is inherently insecure under the CMT3 notion by presenting a simple

<sup>1</sup>In a parallel work, Andreeva *et al.* [AEG<sup>+</sup>26] analyzed the committing security of AE schemes based on the collision-resistance properties of forkcipher.

Table 1: Summary of the committing security of existing forkcipher-based AEAD schemes. The symbols "×" and " $\dagger$ " denote *not secure* and *tight bound*, respectively.

| Construction              | Committing Security |                  |                  |                  | Reference |
|---------------------------|---------------------|------------------|------------------|------------------|-----------|
|                           | CMT1                | CMT2             | CMT3             | CMT4             |           |
| PAEF                      | ×                   | ×                | ×                | ×                | [RKI24]   |
| SAEF                      | ×                   | ×                | ×                | ×                | [RKI24]   |
| Evee Family               | ×                   | ×                | ×                | ×                | [RKI24]   |
| RPAEF (empty message)     | ×                   | ×                | ×                | ×                | Section 3 |
| RPAEF (non-empty message) | $2^{n/2}\dagger$    | $2^{n/2}\dagger$ | ×                | ×                | Section 3 |
| FEDT/FEDT*                | $2^{n/2}\dagger$    | $2^{n/2}\dagger$ | $2^{n/2}\dagger$ | $2^{n/2}\dagger$ | Section 4 |
| ForkDTE1/ForkDTE2         | $2^{n/2}$           | $2^{n/2}$        | $2^{n/2}\dagger$ | $2^{n/2}\dagger$ | Section 5 |

four-query attack. Next, in Section 4, we prove that FEDT and FEDT\* achieves tight  $n/2$ -bit context-committing (CMT4) security. This result shows that, despite earlier negative results, forkcipher-based designs can indeed provide meaningful levels of commitment. Finally, in Section 5, we establish tight  $n/2$ -bit context-committing (CMT4) security for ForkDTE1 and ForkDTE2. These constructions are built upon a forkcipher-based MAC, denoted by ForkMAC. As a key intermediate step, we first design a forkcipher-based hash function, called ForkHash, and prove its collision resistance, which subsequently enables us to establish the committing security of ForkDTE1 and ForkDTE2. All of our security analyses are carried out in the ideal forkcipher model. Table 1 provides a comprehensive overview of the committing security achieved by existing forkcipher-based AEAD schemes across the different notions of committing security.

In fact, our study completes the committing security analysis for all existing forkcipher-based authenticated encryption schemes. Our results further suggest, as an observation, that the forkcipher structure is conducive to achieving committing security, depending on the surrounding mode of operation.

## 2 Preliminaries

**Notation.** For  $n \in \mathbb{N}$ , let  $\{0,1\}^n$  denote the set of binary strings of length  $n$  and  $\{0,1\}^*$  denote the set of all binary strings of arbitrary length. Let  $[q]$  denote the set  $\{1, 2, \dots, q\}$ , for any positive integer  $q$ . An adversary  $\mathcal{A}$  is an algorithm. The notation  $\mathcal{A}(x_1, x_2, \dots, x_i) \rightarrow y$  means that  $\mathcal{A}$  runs on inputs  $x_1, \dots, x_i$  and produces output  $y$ . For a set  $X$ , the notation  $X \stackrel{\cup}{\leftarrow} x$  means that  $x$  is added to  $X$ . For bit strings  $A$  and  $B$ ,  $A\|B$  denotes their concatenation. Let  $|B|$  denote the length (in terms of bits) of a binary string  $B$ . The notation  $[X]_x$  represents the encoding of a non-negative integer  $X < 2^x$  as its  $x$ -bit binary representation. For any set  $X$ ,  $x \stackrel{\$}{\leftarrow} X$  denotes that  $x$  is chosen uniformly at random from  $X$ . We write  $x_1\|x_2\|\dots\|x_\ell \leftarrow_n X$  to denote the parsing of  $X$  into  $\ell$  consecutive blocks, where  $x_i \in \{0,1\}^n$ ,  $\forall i \in [\ell - 1]$  and  $0 < |x_i| \leq n$ . For any  $n$ -bit string  $X$  and two positive integers  $t, s$ ,  $\lceil X \rceil_t$  and  $\lfloor X \rfloor_s$  denote most significant  $t$ -bits and least significant  $s$ -bits of  $X$ .

Let  $\mathcal{M}$  be a finite set. We denote by  $\mathcal{P}(\mathcal{M})$  the set of all permutations over  $\mathcal{M}$ . Let  $\tilde{\mathcal{P}}(\mathcal{T}, \mathcal{M})$  denote the family of all functions  $\tilde{f}: \mathcal{T} \times \mathcal{M} \rightarrow \mathcal{M}$  such that for each  $J \in \mathcal{T}$ , the function  $\tilde{f}(J, \cdot)$  is a permutation on  $\mathcal{M}$ . We write  $\mathcal{P}(n)$  and  $\tilde{\mathcal{P}}(\mathcal{T}, n)$  for the case  $\mathcal{M} = \{0,1\}^n$ .

When an adversary  $\mathcal{A}$  interacts with an oracle  $\mathcal{O}$ , we write  $\mathcal{A}^{\mathcal{O}}$  to denote this interaction. The adversary outputs a bit  $b \in \{0,1\}$ , denoted by  $\mathcal{A}^{\mathcal{O}} \rightarrow b$ . The time complexity of adversaries is measured in the standard RAM model of computation (see, e.g., [Ost90]).

## 2.1 Forkciphers

A tweakable forkcipher  $\tilde{F} = (\tilde{F}^+, \tilde{F}^-)$  is defined over key space  $\mathcal{K}$ , tweak space  $\mathcal{T}$ , and message space  $\{0, 1\}^n$ . It consists of the following deterministic encryption and decryption algorithms.

$$\tilde{F}^+ : \mathcal{K} \times \mathcal{T} \times \{0, 1\}^n \times \{0, 1, 2\} \rightarrow \{0, 1\}^n \cup (\{0, 1\}^n \times \{0, 1\}^n).$$

$$\tilde{F}^- : \mathcal{K} \times \mathcal{T} \times \{0, 1\}^n \times \{0, 1\} \times \{0, 1, 2\} \rightarrow \{0, 1\}^n \cup (\{0, 1\}^n \times \{0, 1\}^n).$$

For  $k \in \mathcal{K}$ ,  $J \in \mathcal{T}$ ,  $m \in \{0, 1\}^n$ , and selector  $s \in \{0, 1, 2\}$ ,

$$\tilde{F}^+(k, J, m, s) = \begin{cases} c_0, & s = 0, \\ c_1, & s = 1, \\ (c_0, c_1), & s = 2, \end{cases}$$

where  $(c_0, c_1)$  denotes the full ciphertext corresponding to  $m$ .

For  $k \in \mathcal{K}$ ,  $J \in \mathcal{T}$ ,  $c_b \in \{0, 1\}^n$ ,  $b \in \{0, 1\}$ , and  $s \in \{0, 1, 2\}$ ,

$$\tilde{F}^-(k, J, c_b, b, s) = \begin{cases} m, & s = 0, \\ c_{1-b}, & s = 1, \\ (m, c_{1-b}), & s = 2, \end{cases}$$

where  $m$  is the unique plaintext such that  $(c_0, c_1) = \tilde{F}^+(k, J, m, 2)$ .

Now, in terms of correctness, for all  $k \in \mathcal{K}$ ,  $J \in \mathcal{T}$ ,  $m \in \{0, 1\}^n$ , and  $b \in \{0, 1\}$ , if  $(c_0, c_1) = \tilde{F}^+(k, J, m, 2)$ , then we have

1.  $\tilde{F}^-(k, J, c_b, b, 0) = m$ ,
2.  $\tilde{F}^-(k, J, c_b, b, 1) = c_{1-b}$ ,
3.  $\tilde{F}^-(k, J, c_b, b, 2) = (\tilde{F}^-(k, J, c_b, b, 0), \tilde{F}^-(k, J, c_b, b, 1))$ ,
4.  $\tilde{F}^+(k, J, m, 2) = (\tilde{F}^+(k, J, m, 0), \tilde{F}^+(k, J, m, 1))$ .

Equivalently, for each key and tweak pair, the two output blocks are obtained by applying two permutations to the input block.

**Ideal Tweakable Forkcipher.** We first define a *tweakable forked permutation* (TFP). Let  $\mathcal{T}$  be a tweak space and let  $n$  denote the block length. For every pair of tweakable permutations  $\tilde{P}_0, \tilde{P}_1 \in \tilde{\mathcal{P}}(\mathcal{T}, n)$ , define

$$\widetilde{FP}[\tilde{P}_0, \tilde{P}_1] = \{\widetilde{FP}^+[\tilde{P}_0, \tilde{P}_1], \widetilde{FP}^-[\tilde{P}_0, \tilde{P}_1]\},$$

where, for every  $J \in \mathcal{T}$ ,  $x, y \in \{0, 1\}^n$ ,  $b \in \{0, 1\}$ , and  $s \in \{0, 1, 2\}$ ,

$$\widetilde{FP}^+[\tilde{P}_0, \tilde{P}_1](J, x, s) = \begin{cases} \tilde{P}_0(J, x), & s = 0, \\ \tilde{P}_1(J, x), & s = 1, \\ (\tilde{P}_0(J, x), \tilde{P}_1(J, x)), & s = 2, \end{cases}$$

and

$$\widetilde{FP}^-[\tilde{P}_0, \tilde{P}_1](J, y, b, s) = \begin{cases} \tilde{P}_b^{-1}(J, y), & s = 0, \\ \tilde{P}_{1 \oplus b}(J, \tilde{P}_b^{-1}(J, y)), & s = 1, \\ (\tilde{P}_b^{-1}(J, y), \tilde{P}_{1 \oplus b}(J, \tilde{P}_b^{-1}(J, y))), & s = 2. \end{cases}$$

Let

$$\widetilde{\mathcal{FP}}(\mathcal{T}, n) = \left\{ \widetilde{\text{FP}}[\widetilde{\text{P}}_0, \widetilde{\text{P}}_1] : \widetilde{\text{P}}_0, \widetilde{\text{P}}_1 \in \widetilde{\mathcal{P}}(\mathcal{T}, n) \right\}$$

denote the collection of all tweakable forked permutations. This collection is viewed as a probability space under the uniform distribution, where sampling  $\widetilde{\text{FP}} \xleftarrow{\$} \widetilde{\mathcal{FP}}(\mathcal{T}, n)$  means independently sampling  $\widetilde{\text{P}}_0, \widetilde{\text{P}}_1 \xleftarrow{\$} \widetilde{\mathcal{P}}(\mathcal{T}, n)$  uniformly at random and setting  $\widetilde{\text{FP}} = \widetilde{\text{FP}}[\widetilde{\text{P}}_0, \widetilde{\text{P}}_1]$ .

Similarly, let  $\text{TFC}(\mathcal{K}, \mathcal{T}, n)$  denote the collection of all tweakable forkciphers with key space  $\mathcal{K}$ , tweak space  $\mathcal{T}$ , and block length  $n$ . Sampling

$$\widetilde{\text{F}} \xleftarrow{\$} \text{TFC}(\mathcal{K}, \mathcal{T}, n)$$

corresponds to independently sampling

$$\forall K \in \mathcal{K} : \quad \widetilde{\text{F}}_K \xleftarrow{\$} \widetilde{\mathcal{FP}}(\mathcal{T}, n).$$

Equivalently, each key defines an independent uniformly random tweakable forked permutation. A tweakable forkcipher generated in this manner is called an *ideal tweakable forkcipher*. This formulation follows [ABCV98].

**Lemma 1.** *Let  $\widetilde{\text{F}}$  be an ideal tweakable forkcipher, and let  $\mathcal{A}$  be an adversary making at most  $q$  oracle queries to  $\widetilde{\text{F}}^\pm$ . Let  $y||z \in \{0, 1\}^{2n}$  be fixed. Then*

$$\Pr \left[ \mathcal{A}^{\widetilde{\text{F}}^\pm} \rightarrow (k, J, x) \text{ s.t. } \widetilde{\text{F}}^+(k, J, x, 2) = y||z \right] \leq \frac{2q}{2^n}.$$

where the oracle query corresponding to the final output appears in the adversary's query transcript.

*Proof.* Fix any target ciphertext pair  $y||z$ . A successful query must satisfy

$$\widetilde{\text{F}}^+(k, J, x, 0) = y \quad \text{and} \quad \widetilde{\text{F}}^+(k, J, x, 1) = z.$$

Since an ideal tweakable forkcipher consists of two independently sampled random tweakable permutations, for any fixed  $(k, J, x)$  the values  $\widetilde{\text{F}}^+(k, J, x, 0)$  and  $\widetilde{\text{F}}^+(k, J, x, 1)$  are independent and uniformly distributed over  $\{0, 1\}^n$ . After at most  $q$  previous queries, at least  $2^n - q$  outputs remain possible for each branch. Hence a fresh forward query matches the prescribed pair  $(y, z)$  with probability at most

$$\frac{1}{(2^n - q)^2} \leq \frac{4}{2^{2n}},$$

where the last inequality holds for  $q \leq 2^{n-1}$ .

Consider an inverse query  $(k, J, z, 1)$ . Since the second branch is a permutation,  $z$  uniquely determines  $x = \widetilde{\text{F}}^-(k, J, z, 1, 0)$ , and the query succeeds only if  $\widetilde{\text{F}}^+(k, J, x, 0) = y$ . After at most  $q$  previous queries, the first branch remains a random permutation over at least  $2^n - q$  unseen outputs. Therefore,

$$\Pr \left[ \widetilde{\text{F}}^+(k, J, x, 0) = y \right] \leq \frac{1}{2^n - q} \leq \frac{2}{2^n},$$

again assuming  $q \leq 2^{n-1}$ . The same argument applies to inverse queries on branch 0.

Hence every query succeeds with probability at most  $2/2^n$ . By the union bound over at most  $q$  queries, we have the bound.  $\square$

**Ideal Forkcipher.** A *forked permutation* (FP) is a tweakable forked permutation over the empty tweak space, i.e.,  $\mathcal{T} = \emptyset$ . Accordingly, we write  $\mathcal{FP}(n)$  for the collection of all forked permutations with block length  $n$ . Sampling  $\text{FP} \xleftarrow{\$} \mathcal{FP}(n)$  corresponds to independently sampling  $P_0, P_1 \xleftarrow{\$} \mathcal{P}(n)$  uniformly at random and setting  $\text{FP} = \text{FP}[P_0, P_1]$ .

Similarly, let  $\text{FC}(\mathcal{K}, n)$  denote the collection of all forkciphers with key space  $\mathcal{K}$  and block length  $n$ . Sampling  $F \xleftarrow{\$} \text{FC}(\mathcal{K}, n)$  corresponds to independently sampling

$$\forall K \in \mathcal{K}, \quad F_K \xleftarrow{\$} \mathcal{FP}(n).$$

Equivalently, each key defines an independent uniformly random forked permutation. A forkcipher generated in this manner is called an *ideal forkcipher*.

## 2.2 AEAD and Committing Security

**Nonce-Based AEAD.** A nonce-based authenticated encryption scheme with associated data  $\text{AE} = (\text{AE}.\mathcal{E}, \text{AE}.\mathcal{D})$  consists of:

- $\text{AE}.\mathcal{E} : \mathcal{K} \times \mathcal{N} \times \mathcal{AD} \times \mathcal{M} \rightarrow \mathcal{C} \times \mathcal{T}$ ,
- $\text{AE}.\mathcal{D} : \mathcal{K} \times \mathcal{N} \times \mathcal{AD} \times \mathcal{C} \times \mathcal{T} \rightarrow \mathcal{M} \cup \{\perp\}$ .

We require correctness: if  $(C, T) = \text{AE}.\mathcal{E}(K, N, A, M)$ , then  $\text{AE}.\mathcal{D}(K, N, A, C, T) = M$ . We say AE is tidy if whenever  $\text{AE}.\mathcal{D}(K, N, A, C, T) = M \neq \perp$ , then  $\text{AE}.\mathcal{E}(K, N, A, M) = (C, T)$ .

### Game CMT1

$(K_1, N_1, A_1, M_1), (K_2, N_2, A_2, M_2) \leftarrow \mathcal{A}^P$   
**if**  $K_1 \neq K_2$   
      $(C_1, T_1) \leftarrow \text{AE}.\mathcal{E}(K_1, N_1, A_1, M_1)$   
      $(C_2, T_2) \leftarrow \text{AE}.\mathcal{E}(K_2, N_2, A_2, M_2)$   
**return**  $((C_1, T_1) = (C_2, T_2)) ? 1 : 0$

### Game CMT2

$(K_1, N_1, A_1, M_1), (K_2, N_2, A_2, M_2) \leftarrow \mathcal{A}^P$   
**if**  $(K_1, N_1) \neq (K_2, N_2)$   
      $(C_1, T_1) \leftarrow \text{AE}.\mathcal{E}(K_1, N_1, A_1, M_1)$   
      $(C_2, T_2) \leftarrow \text{AE}.\mathcal{E}(K_2, N_2, A_2, M_2)$   
**return**  $((C_1, T_1) = (C_2, T_2)) ? 1 : 0$

### Game CMT3

$(K_1, N_1, A_1, M_1), (K_2, N_2, A_2, M_2) \leftarrow \mathcal{A}^P$   
**if**  $(K_1, N_1, A_1) \neq (K_2, N_2, A_2)$   
      $(C_1, T_1) \leftarrow \text{AE}.\mathcal{E}(K_1, N_1, A_1, M_1)$   
      $(C_2, T_2) \leftarrow \text{AE}.\mathcal{E}(K_2, N_2, A_2, M_2)$   
**return**  $((C_1, T_1) = (C_2, T_2)) ? 1 : 0$

### Game CMT4

$(K_1, N_1, A_1, M_1), (K_2, N_2, A_2, M_2) \leftarrow \mathcal{A}^P$   
**if**  $(K_1, N_1, A_1, M_1) \neq (K_2, N_2, A_2, M_2)$   
      $(C_1, T_1) \leftarrow \text{AE}.\mathcal{E}(K_1, N_1, A_1, M_1)$   
      $(C_2, T_2) \leftarrow \text{AE}.\mathcal{E}(K_2, N_2, A_2, M_2)$   
**return**  $((C_1, T_1) = (C_2, T_2)) ? 1 : 0$

Figure 1: Committing security games.

$$\text{CMT4} \iff \text{CMT3} \xrightarrow{\text{green}} \text{CMT2} \xrightarrow{\text{green}} \text{CMT1}$$

Figure 2: Relations among the committing security notions. “ $\iff$ ” denotes equivalency. A green arrow,  $A \xrightarrow{\text{green}} B$ , indicates that security under notion  $A$  implies security under notion  $B$ . Conversely, a red arrow,  $A \xleftarrow{\text{red}} B$ , indicates that an attack under notion  $B$  also constitutes an attack under notion  $A$ .

In the above security games,  $\text{AE}[\text{P}] = (\text{AE}.\mathcal{E}[\text{P}], \text{AE}.\mathcal{D}[\text{P}])$  denotes a nonce-based authenticated encryption scheme with associated data based on an ideal primitive  $\text{P} \xleftarrow{\$} \mathcal{P}$ , to which the adversary is given oracle access. For AEAD committing security, the adversary is required to find two distinct AEAD inputs, including the keys, that encrypt to the same ciphertext-tag pair. We adopt the committing security notions from [BH22] in Figure 1. We write  $\text{Adv}_{\text{AE}}^{\text{Game } G}(\mathcal{A})$  to denote the probability that  $\mathcal{A}$  wins the corresponding game  $G$  when interacting with the ideal primitive  $\text{P}$ . We assume that all queries to  $\text{P}$  needed to evaluate the  $\text{AE}$  on both the pair of final outputs appear in the query transcript of  $\mathcal{A}$ . The advantage is defined as the maximum of the probability of winning over all adversaries making at most  $q_p$  queries to the ideal primitive. We write  $\mathcal{A}^{\text{P}}$  to denote that adversary  $\mathcal{A}$  has oracle access to the ideal primitive  $\text{P}$ . The relations between the committing security notions are illustrated in Figure 2.

### 2.3 Security Notions of Hash Function and MAC

**Collision Resistance.** Let  $\text{H}[\text{P}] : \{0, 1\}^* \rightarrow \{0, 1\}^n$  be a hash function based on an ideal primitive  $\text{P} \xleftarrow{\$} \mathcal{P}$ . The adversary  $\mathcal{A}$  is given oracle access to  $\text{P}$ . The goal of the adversary is to output two distinct messages  $M, M' \in \{0, 1\}^*$  such that

$$M \neq M' \quad \text{and} \quad \text{H}[\text{P}](M) = \text{H}[\text{P}](M').$$

Additionally, we require that all queries to  $\text{P}$  needed to evaluate  $\text{H}[\text{P}](M)$  and  $\text{H}[\text{P}](M')$  appear in the query transcript of  $\mathcal{A}$ .

**Definition 1.** For any adversary  $\mathcal{A}$ , the collision resistance advantage is defined as

$$\text{Adv}_{\text{H}}^{\text{CR}}(\mathcal{A}) = \Pr \left[ \text{P} \xleftarrow{\$} \mathcal{P}, \mathcal{A}^{\text{P}} \rightarrow (M, M') \text{ s.t. } M \neq M', \text{H}[\text{P}](M) = \text{H}[\text{P}](M') \right].$$

A hash function  $\text{H}[\text{P}]$  is said to be  $(\epsilon_{\text{CR}}, q_p)$ -collision resistance hash if, for every adversary  $\mathcal{A}$  making at most  $q_p$  oracle queries to  $\text{P}$ , we have

$$\text{Adv}_{\text{H}}^{\text{CR}}(\mathcal{A}) \leq \epsilon_{\text{CR}}.$$

**Right Collision Resistance.** Let  $\text{H}[\text{P}] : \mathcal{D} \rightarrow \mathcal{X}_l \times \mathcal{X}_r$  be a function based on ideal primitive  $\text{P} \xleftarrow{\$} \mathcal{P}$ . The adversary is given oracle access to  $\mathcal{P}$ .

**Definition 2.** For any adversary  $\mathcal{A}$ , the right collision resistance advantage is defined as

$$\text{Adv}_{\text{H}}^{\text{rcr}}(\mathcal{A}) = \Pr \left[ \text{P} \xleftarrow{\$} \mathcal{P}, \mathcal{A}^{\text{P}} \rightarrow (M, M') \text{ s.t. } \begin{array}{l} M \neq M', \\ \text{H}[\text{P}](M) = (X, Y), \\ \text{H}[\text{P}](M') = (X', Y) \end{array} \right],$$

where all queries needed to evaluate both computations appear in the query transcript of  $\mathcal{A}$ . A function  $\text{H}[\text{P}]$  is  $(\epsilon_{\text{rcr}}, q_p)$ -right collision-resistant if, for every  $\mathcal{A}$  making at most  $q_p$  queries to  $\text{P}$ ,

$$\text{Adv}_{\text{H}}^{\text{rcr}}(\mathcal{A}) \leq \epsilon_{\text{rcr}}.$$

**Preimage Resistance.** We consider the notion of *everywhere preimage resistance* (ePre). An arbitrary target value  $Z \in \{0, 1\}^n$  is fixed, and the goal of the adversary  $\mathcal{A}$  is to output a message  $M$  such that  $\text{H}[\text{P}](M) = Z$ . We require that every invocation of  $\text{P}$  needed to evaluate  $\text{H}[\text{P}](M)$  is included in the query transcript.

**Definition 3.** For any adversary  $\mathcal{A}$ , the everywhere preimage advantage is defined as

$$\text{Adv}_H^{\text{ePre}}(\mathcal{A}) = \max_{Z \in \{0,1\}^n} \Pr \left[ P \xleftarrow{\$} \mathcal{P}, \mathcal{A}^P(Z) \rightarrow M \text{ s.t. } H[P](M) = Z \right].$$

where all queries needed to evaluate the hash appear in the query transcript of  $\mathcal{A}$ . A  $H[P]$  is said to be  $(\epsilon_{\text{ePre}}, q_p)$ -everywhere preimage resistance hash if, for every adversary  $\mathcal{A}$  making at most  $q_p$  oracle queries to  $P$ , we have

$$\text{Adv}_H^{\text{ePre}}(\mathcal{A}) \leq \epsilon_{\text{ePre}}.$$

**Collision Resistance for MACs.** Let  $\text{MAC}[P] : \mathcal{K} \times \mathcal{M} \rightarrow \mathcal{T}$  be a message authentication code based on an ideal primitive  $P \xleftarrow{\$} \mathcal{P}$ . In this work, we are interested only in the collision resistance of MAC.

**Definition 4.** A  $\text{MAC}[P]$  is said to be  $(\epsilon_{\text{maccr}}, q_p)$ -collision resistant (MAC-CR) if, for every adversary  $\mathcal{A}$  making at most  $q_p$  oracle queries to  $P$ ,

$$\Pr \left[ P \xleftarrow{\$} \mathcal{P}, \mathcal{A}^P \rightarrow ((K_0, M_0), (K_1, M_1)) \in (\mathcal{K} \times \mathcal{M})^2 \right. \\ \left. \text{s. t. } (K_0, M_0) \neq (K_1, M_1), \text{MAC}[P](K_0, M_0) = \text{MAC}[P](K_1, M_1) \right] \leq \epsilon_{\text{maccr}}.$$

where we require that all queries to  $P$  needed to evaluate  $\text{MAC}[P](K_0, M_0)$  and  $\text{MAC}[P](K_1, M_1)$  appear in the query transcript of  $\mathcal{A}$ .

### 3 Committing Security of RPAEF

RPAEF [ALP<sup>+</sup>19] is a forkcipher-based authenticated encryption scheme built on the ForkSkinny primitive and originally designed for efficient processing of short messages in constrained environments. A pictorial description and the pseudocode of the encryption algorithm of RPAEF are given in Figure 3 and Algorithm 1, respectively. In this section, we study its committing security and characterize its guarantees under the hierarchy of commitment notions.

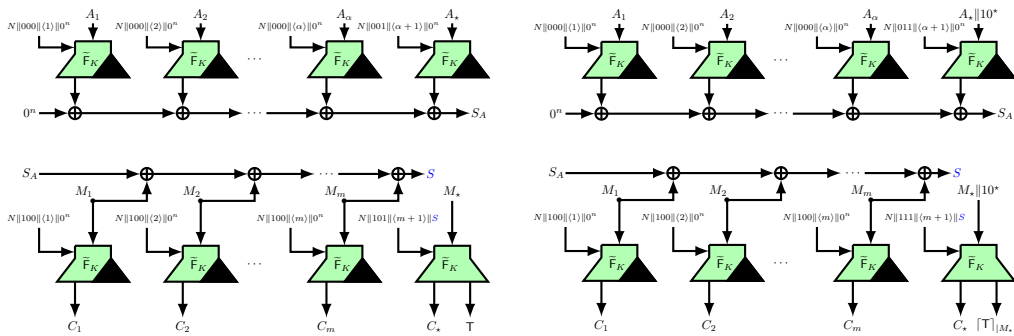


Figure 3: The encryption algorithm of RPAEF[ $\tilde{F}$ ] mode. The picture illustrates the processing of AD when the length of AD is a multiple of  $n$  (**top left**) and when the length of AD is not a multiple of  $n$  (**top right**), and the processing of the message when the length of the message is a multiple of  $n$  (**bottom left**) and when the length of the message is not a multiple of  $n$  (**bottom right**). The black part denotes that an output block is not computed.

**Algorithm 1** Pseudocode of RPAEF $[\tilde{F}, \nu]$ 


---

|                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>function</b> ENC( $K, N, A, M$ )<br>$A_1 \  A_2 \  \dots \  A_\alpha \  A_\star \leftarrow_n A$<br>$M_1 \  M_2 \  \dots \  M_m \  M_\star \leftarrow_n M$<br>$S \leftarrow 0^n; c \leftarrow (t - \nu - 3)$<br><b>for</b> $i \leftarrow 1, \dots, \alpha$ <b>do</b><br>$Tw \leftarrow N \  000 \  \langle i \rangle_c \  0^n$<br>$S \leftarrow S \oplus \tilde{F}_K^{Tw,0}(A_i)$<br><b>if</b> $ A_\star  = n$ <b>then</b><br>$Tw \leftarrow N \  001 \  \langle \alpha + 1 \rangle_c \  0^n$<br>$S \leftarrow S \oplus \tilde{F}_K^{Tw,0}(A_\star)$<br><b>else if</b> $ A_\star  > 0$ or $ M  = 0$ <b>then</b><br>$Tw \leftarrow N \  011 \  \langle \alpha + 1 \rangle_c \  0^n$<br>$S \leftarrow S \oplus \tilde{F}_K^{Tw,0}(A_\star \  10^\star)$<br><b>for</b> $i \leftarrow 1, \dots, m$ <b>do</b><br>$Tw \leftarrow N \  100 \  \langle i \rangle_c \  0^n$<br>$C_i \leftarrow \tilde{F}_K^{Tw,0}(M_i)$<br>$S \leftarrow S \oplus M_i$<br><b>if</b> $ M_\star  = n$ <b>then</b><br>$Tw \leftarrow N \  101 \  \langle m + 1 \rangle_c \  S$<br>$(C_\star, T) \leftarrow \tilde{F}_K^{Tw,2}(M_\star)$<br><b>else if</b> $ M_\star  > 0$ <b>then</b><br>$Tw \leftarrow N \  111 \  \langle m + 1 \rangle_c \  S$<br>$(C_\star, T) \leftarrow \tilde{F}_K^{Tw,2}(M_\star \  10^\star)$<br><b>else</b><br>$\text{return } S$<br>$\text{return } C_1 \  \dots \  C_m \  C_\star \  \lceil T \rceil_{ M_\star }$ | <b>function</b> DEC( $K, N, A, C$ )<br>$A_1 \  A_2 \  \dots \  A_\alpha \  A_\star \leftarrow_n A$<br>$C_1 \  C_2 \  \dots \  C_m \  C_\star \  T \leftarrow_n C$<br>$S \leftarrow 0^n; c \leftarrow (t - \nu - 3)$<br><b>for</b> $i \leftarrow 1, \dots, \alpha$ <b>do</b><br>$Tw \leftarrow N \  000 \  \langle i \rangle_c \  0^n$<br>$S \leftarrow S \oplus \tilde{F}_K^{Tw,0}(A_i)$<br><b>if</b> $ A_\star  = n$ <b>then</b><br>$Tw \leftarrow N \  001 \  \langle \alpha + 1 \rangle_c \  0^n$<br>$S \leftarrow S \oplus \tilde{F}_K^{Tw,0}(A_\star)$<br><b>else if</b> $ A_\star  > 0$ or $ T  = 0$ <b>then</b><br>$Tw \leftarrow N \  011 \  \langle \alpha + 1 \rangle_c \  0^n$<br>$S \leftarrow S \oplus \tilde{F}_K^{Tw,0}(A_\star \  10^\star)$<br><b>for</b> $i \leftarrow 1, \dots, m$ <b>do</b><br>$Tw \leftarrow N \  100 \  \langle i \rangle_c \  0^n$<br>$M_i \leftarrow (\tilde{F}^{-1})_K^{Tw,0,0}(C_i)$<br>$S \leftarrow S \oplus M_i$<br><b>if</b> $ T  = n$ <b>then</b><br>$Tw \leftarrow N \  101 \  \langle m + 1 \rangle_c \  S$<br>$(M_\star, T') \leftarrow (\tilde{F}^{-1})_K^{Tw,0,2}(C_\star)$<br><b>if</b> $T' \neq T$ <b>then</b><br>$\text{return } \perp$<br><b>else if</b> $ T  > 0$ <b>then</b><br>$Tw \leftarrow N \  111 \  \langle m + 1 \rangle_c \  S$<br>$(M_\star, T') \leftarrow (\tilde{F}^{-1})_K^{Tw,0,2}(C_\star)$<br><b>if</b> $\lceil T' \rceil_{ T } \neq T$ <b>then</b><br>$\text{return } \perp$<br><b>if</b> $\lfloor M_\star \rfloor_{n- T } \neq \lceil 10^{n-1} \rceil_{n- T }$ <b>then</b><br>$\text{return } \perp$<br><b>else</b><br><b>if</b> $C_\star \neq S$ <b>then return</b> $\perp$<br>$\text{return } \epsilon$<br>$\text{return } M_1 \  \dots \  M_m \  \lceil M_\star \rceil_{ T }$ |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

---

**3.1 CMT2 Security of RPAEF**

We first present a two-query CMT1 attack (which naturally extends to CMT-2, 3, and 4 due to the hierarchy relation) against RPAEF when the adversary is allowed to output empty messages, denoted by  $\epsilon$ . The attack proceeds as follows.

1. Choose any  $K, N, A_1 \in \{0, 1\}^n$ .
2. Compute  $S_A = \tilde{F}^+(K, N \| 001 \| \langle 1 \rangle \| 0^n, A_1, 0)$ .
3. Choose  $K' \neq K$  and compute  $A'_1 = \tilde{F}^-(K', N \| 001 \| \langle 1 \rangle \| 0^n, S_A, 0, 0)$ .
4. Output the two tuples  $(K, N, A_1, \epsilon)$  and  $(K', N, A'_1, \epsilon)$ .

Both output tuples evaluate to the same internal state  $S_A$ , which serves directly as the final output since the corresponding messages are empty.

Now, if we restrict the adversary to output only non-empty messages, we obtain the following security.

**Theorem 1.** Let  $\tilde{F}$  be an ideal tweakable forkcipher. Let  $\mathcal{A}$  be any CMT2 adversary against RPAEF that is restricted to outputting only non-empty messages, making at most  $q_p$  oracle queries to  $\tilde{F}^+$  and  $\tilde{F}^-$ . Then,

$$\text{Adv}_{\text{RPAEF}}^{\text{CMT2}}(\mathcal{A}) \leq \frac{4q_p^2}{2^n}.$$

*Proof.* Let  $\mathcal{A}$  output two tuples  $(K, N, A, M) \neq (K', N', A', M')$  with  $(K, N) \neq (K', N')$  and  $M, M'$  are non-empty such that

$$\text{RPAEF}(K, N, A, M) = \text{RPAEF}(K', N', A', M'),$$

Let the corresponding outputs of RPAEF be  $(C_1 \| \dots \| C_m \| C_\star, T) = (C'_1 \| \dots \| C'_{m'} \| C'_\star, T')$ .

Now, if exactly one of the two message lengths is divisible by  $n$ , then the corresponding tags have different lengths, and hence the outputs cannot be equal. Therefore, we only consider the cases where both message lengths are divisible by  $n$  or both are not divisible by  $n$ . The final ciphertext block and tag are generated by the final forkcipher call:

$$(C_\star, T) = \tilde{F}_K^{Tw, 2}(X), \quad (C'_\star, T') = \tilde{F}_{K'}^{Tw', 2}(X'),$$

where  $Tw = N \| d \| \langle m+1 \rangle_c \| S$ , with  $d = 101$  for a complete last block and  $d = 111$  for a partial last block. Since  $(K, N) \neq (K', N')$ , we have  $(K, Tw) \neq (K', Tw')$ , and thus the two final forkcipher calls correspond to distinct inputs.

**Case 1. The message lengths are multiples of  $n$ :** Let  $M_\star$  and  $M'_\star$  denote the final message blocks. In this case,

$$(C_\star, T) = \tilde{F}^+(K, Tw, M_\star, 2), \quad (C'_\star, T') = \tilde{F}^+(K', Tw', M'_\star, 2).$$

For a successful collision, the adversary must find two distinct tweakable forkcipher inputs satisfying

$$\tilde{F}^+(K, Tw, M_\star, 2) = \tilde{F}^+(K', Tw', M'_\star, 2).$$

Assume, without loss of generality, that the second query is issued after the first one. For a fixed output pair generated by the first query, the probability of finding a second query that produces the same pair is at most  $2q_p/2^n$  by Lemma 1. Since there are at most  $q_p$  possible target output pairs, the success probability is at most  $2q_p^2/2^n$ .

**Case 2. The message lengths are not multiples of  $n$ :** If the forkcipher call corresponding to  $(K', Tw')$  is a forward call, i.e.,  $\tilde{F}^+(K', Tw', M'_\star \| 10^\star, 2)$ , then the adversary must satisfy

$$C'_\star = C_\star \text{ and } \lceil T' \rceil_{|M'_\star|} = \lceil T \rceil_{|M_\star|}.$$

For a fixed target  $(C_\star, \lceil T \rceil_{|M_\star|})$ , this occurs with probability at most  $1/2^{n+|M_\star|}$ . Since there are at most  $q_p$  possible targets and  $q_p$  possible queries, the success probability of the adversary is upper bounded by  $q_p^2/2^{n+|M_\star|}$ .

Otherwise, the adversary may attempt to find suitable  $K'$ ,  $Tw'$ , and  $M'_\star$  corresponding to  $(C_\star, \lceil T \rceil_{|M_\star|})$  via inverse queries. In this case, the adversary must also match the padding sequence  $10^\star$  in the recovered message part and ensure correctness of the pair  $(C_\star, \lceil T \rceil_{|M_\star|})$ . Since there are at most  $q_p$  possible targets and  $q_p$  possible queries, and the adversary must satisfy an  $n$ -bit constraint, where  $n = |\lceil T \rceil_{|M_\star|}| + |10^\star|$ , the success probability is upper bounded by  $q_p^2/2^n$ .

Combining the above cases, the result follows.  $\square$

**Tightness of the CMT2 security of RPAEF.** We now demonstrate the tightness of the CMT2 bound for RPAEF established in the preceding theorem. The attack proceeds as follows.

1. Choose arbitrary values  $N, S, C_\star \in \{0, 1\}^n$  and  $2^{n/2}$  distinct keys  $K^\ell$ .
2. For each  $\ell \in [2^{n/2}]$ , compute  $(M_\star^\ell, T^\ell) = \tilde{F}^-(K^\ell, N \| 101 \| \langle 2 \rangle \| S, C_\star, 0, 2)$ .
3. Find indices  $i, j \in [2^{n/2}]$  such that  $T^i = T^j$ .
4. Choose any  $C_1 \in \{0, 1\}^n$ , and compute  $M_1^i = \tilde{F}^-(K^i, N \| 100 \| \langle 1 \rangle \| 0^n, C_1, 0, 0)$  and  $M_1^j = \tilde{F}^-(K^j, N \| 100 \| \langle 1 \rangle \| 0^n, C_1, 0, 0)$ .
5. Compute  $S_A^i = S \oplus M_1^i$  and  $S_A^j = S \oplus M_1^j$ .
6. Choose any value  $S_2 \in \{0, 1\}^n$ . Compute  $A_2^i = \tilde{F}^-(K^i, N \| 001 \| \langle 2 \rangle \| 0^n, S_2, 0, 0)$  and  $A_2^j = \tilde{F}^-(K^j, N \| 001 \| \langle 2 \rangle \| 0^n, S_2, 0, 0)$ .
7. Compute  $S_1^i = S_2 \oplus S_A^i$  and  $A_1^i = \tilde{F}^-(K^i, N \| 000 \| \langle 1 \rangle \| 0^n, S_1^i, 0, 0)$ .
8. Compute  $S_1^j = S_2 \oplus S_A^j$  and  $A_1^j = \tilde{F}^-(K^j, N \| 000 \| \langle 1 \rangle \| 0^n, S_1^j, 0, 0)$ .
9. Output the two tuples  $(K^i, N, A_1^i \| A_2^i, M_1^i \| M_\star^i)$  and  $(K^j, N, A_1^j \| A_2^j, M_1^j \| M_\star^j)$ .

Since the values  $T^\ell$  are independent and uniformly random over  $\{0, 1\}^n$ , the attack succeeds with a non-negligible probability by the birthday paradox. From the relationship between the CMT1 and CMT2 notions, we can conclude that RPAEF achieves at least the same level of CMT1 security as it does CMT2 security. Furthermore, the attack described above constitutes a valid CMT1 attack. Therefore, it not only demonstrates the tightness of the CMT2 security bound, but also establishes that the corresponding CMT1 security bound is tight.

### 3.2 CMT3 Attack on RPAEF

Now we demonstrate a four-query CMT3 attack on RPAEF. Our goal is to find two distinct tuples,  $(K, N, A_1 \| A_2, M)$  and  $(K, N, A'_1 \| A'_2, M)$  such that

$$\text{RPAEF}(K, N, A_1 \| A_2, M) = \text{RPAEF}(K, N, A'_1 \| A'_2, M).$$

The detailed attack algorithm is as follows.

1. Choose any  $K, N, A_1 \| A_2, M$ .
2. Compute  $S_1 = \tilde{F}^+(K, N \| 000 \| \langle 1 \rangle \| 0^n, A_1, 0)$  and  $S_2 = \tilde{F}^+(K, N \| 001 \| \langle 2 \rangle \| 0^n, A_2, 0)$ .
3. Compute  $S_A = S_1 \oplus S_2$ .
4. Choose  $(S'_1, S'_2) \neq (S_1, S_2)$  such that  $S'_1 \oplus S'_2 = S_1 \oplus S_2$ .
5. Compute  $A'_1 = \tilde{F}^-(K, N \| 000 \| \langle 1 \rangle \| 0^n, S'_1, 0, 0)$ .
6. Compute  $A'_2 = \tilde{F}^-(K, N \| 001 \| \langle 2 \rangle \| 0^n, S'_2, 0, 0)$ .
7. Output the two tuples  $(K, N, A_1 \| A_2, M)$  and  $(K, N, A'_1 \| A'_2, M)$ .

**Correctness of the attack.** The construction of RPAEF proceeds in two stages. In the first stage, the associated data is processed using the key and nonce to produce a value  $S_A$ . In the second stage, this value  $S_A$ , together with the same key and nonce and the message, is used to generate the ciphertext and tag.

Therefore, if one can obtain a collision in  $S_A$  for the same key and nonce by modifying only the associated data, then the resulting ciphertext-tag pair will also collide. In the attack above, we enforce

$$S'_A = S'_1 \oplus S'_2 = S_1 \oplus S_2 = S_A.$$

The values  $A'_1$  and  $A'_2$  are then constructed to realize this same  $S'_A$ . Consequently, for the same message  $M$ , the encryption algorithm produces identical ciphertext-tag pairs. Hence, the adversary succeeds with probability 1.

From the relationship between the CMT3 and CMT4 notions, we can conclude that the attack described above constitutes a valid CMT4 attack also.

## 4 CMT4 Security of FEDT

The first tweakable forkcipher-based leakage-resilient AEAD schemes FEDT and FEDT\* were introduced in [DDLM24]. Both constructions follow the Encrypt-then-MAC paradigm and rely on two strongly protected tweakable forkciphers instantiated under a single master key. The two schemes differ only in their encryption procedures: FEDT employs a balanced tree structure to enable parallel encryption, whereas FEDT\* adopts an unbalanced tree to reduce latency while providing comparable security guarantees.

In this section, we establish the CMT4 security of FEDT. Our proof is entirely modular and combines three existing results. We first reduce the committing security of the AEAD scheme to the collision resistance of its underlying MAC. Next, we observe that the MAC used by FEDT is an instance of the generic LRMAC1 construction [BGPS21]. Finally, we study the generic LRMAC1 analysis using the collision resistance of the tweakable forkcipher-based hash employed by FEDT. Since FEDT and FEDT\* share the same MAC component, the committing security (CMT4) analysis immediately applies to both schemes.

**Definition 5.** Let  $\Pi$  be a nonce-based AEAD scheme consisting of three algorithms  $\text{KeyGen}$ ,  $\text{Enc}$ , and  $\text{Mac}$ , such that the encryption of  $\Pi$  is computed as

$$(K_e, K_m) \leftarrow \text{KeyGen}(K), \quad C \leftarrow \text{Enc}(K_e, N, M), \quad T \leftarrow \text{Mac}(K_m, N, A, C).$$

Then  $\Pi$  is said to follow the *Encrypt-then-MAC* (EtM) paradigm.

Both FEDT and FEDT\* satisfy Definition 5. Their MAC component first computes an unkeyed tweakable forkcipher-based hash of the nonce, associated data, and ciphertext, and subsequently evaluates a tweakable forkcipher under the master key using the hash value as the tweak. Concretely, the procedure of FEDT is given by

$$\begin{cases} C \leftarrow \text{Enc}[\tilde{F}](K, N, M), \\ U \| V \leftarrow \text{H}[\tilde{F}](A, N, C), \\ T \leftarrow \tilde{F}^+(K, U \| V, 0^n, 0). \end{cases}$$

The tweakable forkcipher-based hash function used by FEDT is specified in Algorithm 2, and the overall construction is illustrated in Figure 4.

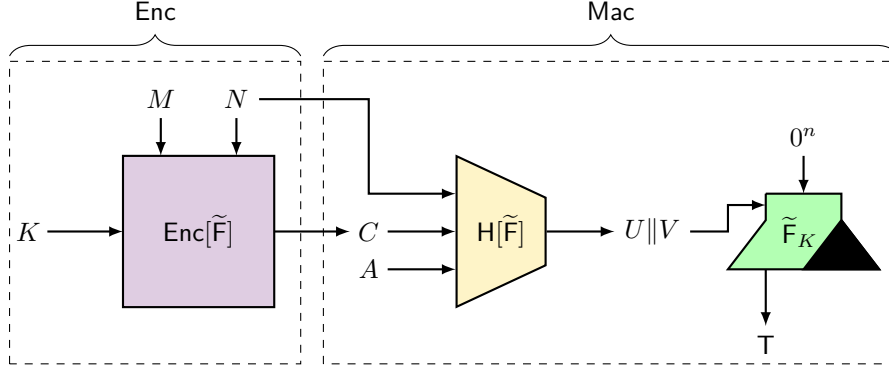


Figure 4: Encryption of FEDT.

**Algorithm 2** Specification of FEDT-hash

|                                                                                                                                                                                                                                                                                                                                      |                                                                                                                                                                                                    |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>function</b> $H[\tilde{F}](A, N, C)$<br>$U \leftarrow \text{pad}(A, N, C)$<br>$U_1    U_2    \dots    U_l \leftarrow_{2n} U$<br>$u_0 \leftarrow 0^n$<br>$v_0 \leftarrow 0^n$<br><b>for</b> $i \leftarrow 1, 2, \dots, l$ <b>do</b><br>$u_i    v_i \leftarrow \tilde{F}^+(v_{i-1}, U_i, u_{i-1}, 2)$<br><b>return</b> $u_l    v_l$ | <b>function</b> $\text{pad}(A, N, C)$<br>$l \leftarrow  A  C $<br>$\delta \leftarrow l \bmod 2n$<br>$Y \leftarrow (A    C    0^{2n-\delta})    N    ([A]_{n/2}    [C]_{n/2})$<br><b>return</b> $Y$ |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

**Reducing CMT4 security to collision resistance of MAC.** Since FEDT follows the Encrypt-then-MAC paradigm, its committing security can be analyzed modularly through the collision resistance of its MAC component. The following theorem, due to Dhar et al. [DEJ<sup>+</sup>24], provides such a reduction for arbitrary Encrypt-then-MAC constructions.

**Lemma 2** ([DEJ<sup>+</sup>24, Theorem 1]). Let  $\Pi$  be an AEAD scheme based on the EtM paradigm that consists of the key generation algorithm  $\text{KeyGen}$ , the encryption algorithm  $\text{Enc}$ , and a MAC algorithm  $\text{MAC}$ . Suppose that  $\text{KeyGen}$  is  $(\epsilon_{\text{rcr}}, q_p)$ -right collision-resistant and  $\text{MAC}$  is  $(\epsilon_{\text{maccr}}, q_p)$ -collision-resistant. Then, for every CMT4 adversary  $\mathcal{A}$  making at most  $q_p$  oracle queries to the underlying ideal primitive  $P$ ,

$$\text{Adv}_{\Pi}^{\text{CMT4}}(\mathcal{A}) \leq \epsilon_{\text{maccr}} + \epsilon_{\text{rcr}}.$$

We now specialize the above result to FEDT. In both FEDT and FEDT\*, the encryption and MAC procedures use the same master key, that is,  $K_e = K_m$ . Consequently, the right collision resistance term vanishes. In particular, [DEJ<sup>+</sup>24, Theorem 1] shows that  $\epsilon_{\text{rcr}} = 0$ . Therefore,

$$\text{Adv}_{\text{FEDT}}^{\text{CMT4}}(\mathcal{A}) \leq \epsilon_{\text{maccr}}.$$

Hence, establishing the CMT4 security of FEDT reduces to proving the collision resistance of its MAC component. Since FEDT\* employs the same authentication mechanism, the same reduction immediately applies to FEDT\*. In the remainder of this section, we focus exclusively on the MAC used by FEDT.

**Collision resistance of the MAC of FEDT.** By the reduction established in Lemma 2, it remains to analyze the collision resistance of the MAC used by FEDT. Recall that the MAC component first computes  $U || V \leftarrow H[\tilde{F}](A, N, C)$ , and then evaluates the tweakable

forkcipher under the master key using the hash value as the tweak and the fixed plaintext  $0^n$ , i.e.,

$$T \leftarrow \tilde{F}^+(K, U \| V, 0^n, 0).$$

Since only one branch of the forkcipher output is used during tag generation, this invocation behaves exactly as a tweakable block cipher (TBC). Consequently, the MAC employed by FEDT is an instance of the LRMAC1 framework introduced in [BGPS21]. The LRMAC1 construction is defined as follows.

**Definition 6.** Let  $H : \mathcal{M} \rightarrow \{0, 1\}^\tau$  be a hash function and  $\tilde{E} : \mathcal{K}_m \times \{0, 1\}^\tau \times \{0, 1\}^n \rightarrow \{0, 1\}^n$  be a tweakable block cipher. For a key  $K_m$  and a message  $M$ , the construction  $\text{LRMAC1}[H, \tilde{E}]$  outputs

$$\text{LRMAC1}[H, \tilde{E}](K_m, M) = \tilde{E}(K_m, H(M), 0^n).$$

Since the tag generation algorithm of FEDT follows exactly this structure, we may directly invoke the generic security analysis of LRMAC1.

**Lemma 3** ([DEJ<sup>+</sup>24, Theorem 4]). Let  $\tilde{E} : \mathcal{K}_m \times \{0, 1\}^\tau \times \{0, 1\}^n \rightarrow \{0, 1\}^n$  be an ideal tweakable block cipher and  $H : \mathcal{M} \rightarrow \{0, 1\}^\tau$  be a  $(\epsilon_{\text{CR}}, q_p)$ -collision-resistant hash function. Then, for every adversary  $\mathcal{A}$  making at most  $q_p \leq 2^{n-1}$  queries to the ideal tweakable block cipher,  $\text{LRMAC1}[H, \tilde{E}]$  is  $(\epsilon_{\text{maccr}}, q_p)$ -collision-resistant for

$$\epsilon_{\text{maccr}} \leq \epsilon_{\text{CR}} + \frac{q_p^2 + 2q_p + 5}{2^n}.$$

Lemma 3 reduces the collision resistance of the MAC to that of the underlying hash function. Therefore, the only remaining ingredient required for the analysis of FEDT is a collision-resistance bound for the specific tweakable forkcipher-based hash function  $H[\tilde{F}]$ . Although hash and tag generation use the same ideal tweakable forkcipher, we can directly apply [DEJ<sup>+</sup>24, Theorem 4] to our setting, since the final tweakable block cipher is evaluated only on the fixed plaintext  $0^n$ .

**Collision Resistance of the Underlying Hash.** By Lemma 3, it remains only to bound the collision resistance of the tweakable forkcipher-based hash function  $H[\tilde{F}]$  used by FEDT. This hash function is described in Algorithm 2. Its collision resistance was established in [DDL24], which we restate below for completeness.

**Lemma 4** ([DDL24, Lemma 2]). Let  $\tilde{F}$  be an ideal tweakable forkcipher. Then, for every adversary  $\mathcal{A}$  making at most  $q_p \leq 2^{n-1}$  oracle queries to  $\tilde{F}$ , the hash function  $H[\tilde{F}]$  is  $(\epsilon_{\text{CR}}, q_p)$ -collision-resistant, where

$$\epsilon_{\text{CR}} \leq \frac{q_p}{2^n} + \frac{q_p^2}{2^{2n}} + \frac{q_p^3}{2^{3n}}.$$

Lemma 4 provides exactly the collision-resistance bound required by Lemma 3. We are now ready to combine the three ingredients established in this section to derive the CMT4 security of FEDT.

**Theorem 2.** Let  $\tilde{F}$  be an ideal tweakable forkcipher. Then, for every CMT4 adversary  $\mathcal{A}$  making at most  $q_p \leq 2^{n-1}$  oracle queries to  $\tilde{F}^+$  and  $\tilde{F}^-$ ,

$$\text{Adv}_{\text{FEDT}}^{\text{CMT4}}(\mathcal{A}) \leq \frac{q_p^2 + 3q_p + 5}{2^n} + \frac{q_p^2}{2^{2n}} + \frac{q_p^3}{2^{3n}}.$$

*Proof.* Following Lemma 2 and the fact that FEDT uses the same master key for encryption and authentication, we have

$$\begin{aligned} \text{Adv}_{\text{FEDT}}^{\text{CMT4}}(\mathcal{A}) &\leq \epsilon_{\text{maccr}} + 0 \leq \frac{q_p}{2^n} + \frac{q_p^2}{2^{2n}} + \frac{q_p^3}{2^{3n}} + \frac{q_p^2 + 2q_p + 5}{2^n} \\ &= \frac{q_p^2 + 3q_p + 5}{2^n} + \frac{q_p^2}{2^{2n}} + \frac{q_p^3}{2^{3n}}, \end{aligned}$$

where we employ Lemma 3 and Lemma 4 in the second inequality.  $\square$

**Tightness of the CMT4 security of FEDT.** We now demonstrate the tightness of the CMT4 bound for FEDT established in the preceding theorem. The attack proceeds as follows.

1. Choose arbitrary values  $N, A, C$  and  $2^{n/2}$  distinct keys  $K_\ell$ .
2. Compute  $U \| V = \text{H}[\tilde{\text{F}}](A, N, C)$  and  $\text{T}_\ell = \tilde{\text{F}}^+(K_\ell, U \| V, 0^n, 0)$ ,  $\forall \ell \in [2^{n/2}]$ .
3. Find indices  $i, j \in [2^{n/2}]$  such that  $\text{T}_i = \text{T}_j$ .
4. Compute  $M_i = \text{Dec}[\tilde{\text{F}}](K_i, N, C)$  and  $M_j = \text{Dec}[\tilde{\text{F}}](K_j, N, C)$ .
5. Output the two tuples  $(K_i, N, A, M_i)$  and  $(K_j, N, A, M_j)$ .

Since the values  $\text{T}_\ell$ 's are independent and uniformly random over  $\{0, 1\}^n$ , the attack succeeds with a non-negligible probability by the birthday paradox. From the relationship between the CMT1 to CMT4 notions, we can conclude that FEDT achieves at least the same level of CMT1, CMT2, and CMT3 security as it does CMT4 security. Observe that this attack is, in fact, a CMT1 attack. Consequently, it not only establishes the tightness of the CMT4 bound, but also shows that the corresponding CMT1, CMT2, and CMT3 bounds are tight.

## 5 CMT4 security of ForkDTE1 (and ForkDTE2)

In [BSL24], the authors propose a MAC construction ForkMAC based on a forkcipher  $\text{F}$  and a hash function  $\text{H}$ . The MAC is defined as

$$\begin{aligned} \text{ForkMAC} : \{0, 1\}^\kappa \times \{0, 1\}^* &\rightarrow \{0, 1\}^n, \\ \text{ForkMAC}(K, M) &:= \text{F}^+(K, \text{H}(M), 0) = \text{T}. \end{aligned}$$

They show that this construction achieves strong existential unforgeability under chosen-message and verification attacks in the presence of leakage, in the sense of [BGP<sup>+</sup>19]. This security guarantee relies on the collision resistance of the underlying hash function  $\text{H}$ .

Building on this construction, and following the design paradigm of DTE [BKP<sup>+</sup>16] and DTE2 [BPPS17], the authors extend the MAC to two authenticated encryption schemes, denoted by ForkDTE1 and ForkDTE2, using an encryption layer inspired by PSV [PSV15]. Both schemes employ the same encryption algorithm and differ only in the decryption procedure, specifically in the method used for tag verification.

Since our focus is on committing security, we restrict attention to the encryption algorithm. Indeed, committing security depends solely on the encryption process and is independent of the verification procedure used in decryption.

Let  $\text{H}$  be a hash function,  $\text{F}$  be a forkcipher, and  $\text{PSV\_Enc}$  be the encryption algorithm defined in [BSL24]. Then, the encryption procedure for both ForkDTE1 and ForkDTE2 is given by

$$h \leftarrow \text{H}(N, A, M), \quad (\text{T}, K_0) \leftarrow \text{F}^+(K, h, 2), \quad C \leftarrow \text{PSV\_Enc}(K_0, N, M),$$

and the output is  $(C, T)$ . Observe that  $T$  is the output of  $\text{ForkMAC}(K, N, A, M)$ . A schematic diagram of the encryption procedure for ForkDTE1 and ForkDTE2 with associated data is shown in Figure 5.

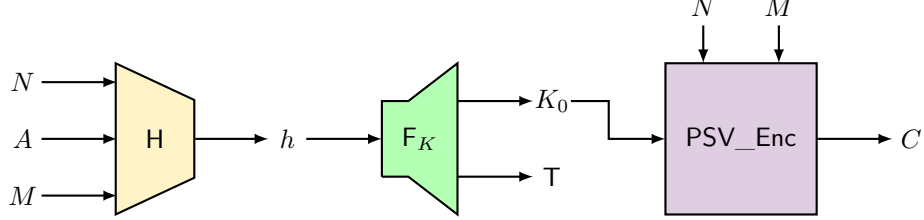


Figure 5: Encryption of ForkDTE1 and ForkDTE2.

We note that any successful adversary producing two distinct inputs that result in the same ciphertext-tag pair necessarily induces a collision in the underlying MAC ForkMAC. Therefore, the committing security of ForkDTE1 (and similarly ForkDTE2) reduces to the collision resistance of ForkMAC.

**Theorem 3.** *Let  $F$  be an ideal forkcipher. Let  $\mathcal{A}$  be an adversary against ForkDTE1 in the CMT4 game making at most  $q_p$  oracle queries to  $F^+$  and  $F^-$ . Then there exists an adversary  $\mathcal{A}'$  such that*

$$\text{Adv}_{\text{ForkDTE1}}^{\text{CMT4}}(\mathcal{A}) \leq \text{Adv}_{\text{ForkMAC}}^{\text{CR}}(q_p) \leq \frac{16q_p^2}{2^n} + \frac{4q_p}{2^n}.$$

*Proof.* We prove the result by reducing to the collision resistance of ForkMAC. Let  $\mathcal{A}'$  be an adversary that attempts to find collisions for ForkMAC. The adversary  $\mathcal{A}'$  has access to an ideal forkcipher  $F^\pm$ .

The adversary  $\mathcal{A}'$  internally runs  $\mathcal{A}$  and simulates its environment as follows. For each query made by  $\mathcal{A}$ ,  $\mathcal{A}'$  forwards the corresponding forkcipher queries to its own oracle  $F^\pm$  and returns the responses to  $\mathcal{A}$ .

Eventually,  $\mathcal{A}$  outputs two distinct tuples  $(K, N, A, M)$  and  $(K', N', A', M')$ . The adversary  $\mathcal{A}'$  outputs these two tuples as its candidate collision. If  $\mathcal{A}$  wins the CMT4 game against ForkDTE1, then we have

$$(K, N, A, M) \neq (K', N', A', M') \text{ s.t. } \text{ForkDTE1}(K, N, A, M) = \text{ForkDTE1}(K', N', A', M').$$

By the structure of ForkDTE1, this implies a collision in the underlying ForkMAC computation. Hence,  $\mathcal{A}'$  successfully finds a collision for ForkMAC.

Therefore, the advantage of  $\mathcal{A}$  in the CMT4 game is upper bounded by the collision advantage against ForkMAC from Theorem 5, which completes the proof.  $\square$

**Theorem 4.** *Let  $F$  be an ideal forkcipher. Let  $\mathcal{A}$  be an adversary against ForkDTE2 in the CMT4 game making at most  $q_p$  oracle queries to  $F^+$  and  $F^-$ . Then there exists an adversary  $\mathcal{A}'$  such that*

$$\text{Adv}_{\text{ForkDTE2}}^{\text{CMT4}}(\mathcal{A}) \leq \text{Adv}_{\text{ForkMAC}}^{\text{CR}}(q_p) \leq \frac{16q_p^2}{2^n} + \frac{4q_p}{2^n}.$$

*Proof.* The proof is exactly similar to the proof of Theorem 3, because ForkDTE2 uses the same encryption algorithm as ForkDTE1.  $\square$

**Collision Resistance of ForkMAC.** We first describe a forkcipher-based hash function, denoted by ForkHash (or H), constructed following the Merkle–Damgård paradigm using a forkcipher-based compression function  $C : \{0,1\}^n \times \{0,1\}^n \rightarrow \{0,1\}^n$ . The detailed construction of ForkMAC based on ForkHash is given in Algorithm 3 and a diagram in Figure 6.

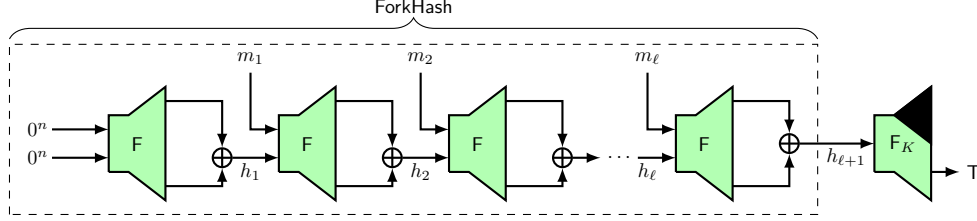


Figure 6: ForkMAC based on ForkHash.

We begin by proving the preimage resistance and collision resistance of the compression function  $C$ . Building on these results, we establish the collision resistance of the hash function ForkHash. Finally, we use this to prove the collision resistance of ForkMAC.

---

**Algorithm 3** Specification of ForkMAC based on ForkHash.

---

**function**

ForkMAC[H, F](K, N, A, M)

 $h \leftarrow H(N, A, M)$  $T \leftarrow F^+(K, h, 0)$ **return** T**function** H(N, A, M) $U \leftarrow \text{pad}(N, A, M)$  $m_1 \| m_2 \| \dots \| m_\ell \leftarrow_n U$  $h_0 \leftarrow 0^n$  $m_0 \leftarrow 0^n$ **for**  $i \leftarrow 1, 2, \dots, \ell + 1$  **do** $h_i \leftarrow C(h_{i-1}, m_{i-1})$ **return**  $h_{\ell+1}$ **function** C(k, m) $z \leftarrow F^+(k, m, 0) \oplus F^+(k, m, 1)$ **return** z**function** pad(N, A, M) $l \leftarrow |A| \| |M|$  $\delta \leftarrow l \bmod n$  $Y \leftarrow N \| A \| M \| 0^{n-\delta} \| [|A|]_{n/2} \| [|M|]_{n/2}$ **return** Y

**Lemma 5** (Preimage resistance of C). *Let F be an ideal forkcipher and  $\mathcal{A}$  be an adversary issuing at most  $q_p \leq 2^{n-1}$  queries to the forkcipher oracle  $F^\pm$ . Then, we have*

$$\text{Adv}_C^{\text{ePre}}(\mathcal{A}) \leq \frac{4q_p}{2^n}.$$

*Proof.* Note that, by definition of the compression function, for any  $z \in \{0,1\}^n$  we have

$$C(k, m) = z \Leftrightarrow F^+(k, m, 2) = c_0 \| c_1 \text{ s.t. } c_0 \oplus c_1 = z.$$

Finding  $(k, m)$  for the C function is equivalent to finding  $(k, m)$  such that  $F^+(k, m, 2) = c_0 \| c_1$  and  $c_0 \oplus c_1 = z$ .

For any  $z \in \{0,1\}^n$ , there are exactly  $2^n$  pairs  $(c_0, c_1) \in \{0,1\}^n \times \{0,1\}^n$  such that  $c_0 \oplus c_1 = z$ . Moreover, for any fixed  $c_0$ , there exists a unique  $c_1 = z \oplus c_0$ .

Now,  $\mathcal{A}$  can make forward and backward queries to the forkcipher oracle. If  $\mathcal{A}$  makes a forward call with  $(k, m)$ , then from the ideal forkcipher property, we have

$$\Pr[F^+(k, m, 2) = c_0 \| c_1 \text{ s.t. } c_0 \oplus c_1 = z] \leq \frac{2^n}{(2^n - q_p) \cdot (2^n - q_p)} \leq \frac{4}{2^n}.$$

If  $\mathcal{A}$  makes a backward query with  $(k, c_b, b, s)$ , then  $c_{1-b}$  is uniquely determined by the forkcipher permutation corresponding to key  $k$  and the unknown preimage of  $c_b$ . Hence, consistency with the condition  $c_0 \oplus c_1 = z$  requires that this uniquely induced value satisfies  $c_{1-b} = z \oplus c_b$ . Since at most  $q_p$  input-output pairs have been fixed by previous queries, the probability that this consistency condition holds is bounded by

$$\Pr[c_{1-b} = z \oplus c_b] \leq \frac{1}{2^n - q_p} \leq \frac{2}{2^n}.$$

Hence, each oracle query contributes at most  $\frac{4}{2^n}$  to the success probability of producing a valid preimage  $(k, m)$  such that  $C(k, m) = z$ . Since  $\mathcal{A}$  makes at most  $q_p$  queries, by a union bound we obtain

$$\Pr[\mathcal{A}^{\mathbf{F}^\pm} \rightarrow (k, m) \text{ s.t. } C(k, m) = z] \leq \frac{4q_p}{2^n}.$$

□

**Lemma 6** (Collision resistance of  $C$ ). *Let  $F$  be an ideal forkcipher and  $\mathcal{A}$  be an adversary making at most  $q_p \leq 2^{n-1}$  queries to  $F^\pm$ . Then, we have*

$$\text{Adv}_C^{\text{CR}}(\mathcal{A}) \leq \frac{4q_p^2}{2^n}.$$

*Proof.* Recall that  $C(k, m) = z$  iff  $F^+(k, m, 2) = (c_0, c_1)$  and  $z = c_0 \oplus c_1$ . Thus, a collision occurs if the adversary produces two distinct queries  $(k, m)$  and  $(k', m')$  such that

$$c_0 \oplus c_1 = c'_0 \oplus c'_1,$$

where  $F^+(k, m, 2) = (c_0, c_1)$  and  $F^+(k', m', 2) = (c'_0, c'_1)$ .

Now, consider any two distinct queries  $(k, m)$  and  $(k', m')$  made by  $\mathcal{A}$ , where without loss of generality  $(k', m')$  is issued after  $(k, m)$  in the adversary's execution. A collision occurs only if

$$C(k', m') = C(k, m).$$

Fixing  $z = C(k, m)$ , the adversary succeeds only if the later query  $(k', m')$  is a preimage of  $z$  under the compression function.

By Lemma 5, for any fixed  $z \in \{0, 1\}^n$ , the probability that a fresh query  $(k', m')$  satisfies  $C(k', m') = z$  is at most  $\frac{4q_p}{2^n}$ .

Since there are at most  $q_p$  choices for the earlier query  $(k, m)$ , applying a union bound yields the lemma. □

**Lemma 7** (Fixed-point resistance of  $C$ ). *Let  $F$  be an ideal forkcipher and  $\mathcal{A}$  be an adversary making at most  $q_p \leq 2^{n-1}$  queries to the ideal forkcipher oracle  $F^\pm$ . Then,*

$$\Pr[\mathcal{A}^{\mathbf{F}^\pm} \rightarrow (K, x) \text{ s.t. } C(K, x) = x] \leq \frac{4q_p}{2^n}.$$

*Proof.* The adversary succeeds only if it finds a query  $(K, x)$  such that, for  $(c_0, c_1) = F^+(K, x, 2)$ ,  $c_0 \oplus c_1 = x$ . We analyze the probability of the event that any forkcipher query satisfies this condition.

Consider first a forward query  $(K, x)$ . For a fresh input  $(K, x)$ , the pair  $(c_0, c_1)$  is uniformly distributed over  $\{0, 1\}^{2n}$ , conditioned on the previous queries. For any  $x$ , there are exactly  $2^n$  pairs  $(c_0, c_1)$  satisfying  $c_0 \oplus c_1 = x$ . Hence, after at most  $q_p$  previous queries, the probability of satisfying the condition is upper bounded by

$$\frac{2^n}{(2^n - q_p)^2} \leq \frac{4}{2^n}.$$

Now consider an inverse query  $(K, c_b, b, 0)$ . The queried branch uniquely determines the corresponding plaintext  $x$  and the other ciphertext branch  $c_{1-b}$ . Thus, the condition  $c_0 \oplus c_1 = x$  is equivalent to  $c_{1-b} = x \oplus c_b$ . Conditioning on the previous queries, the value  $c_{1-b}$  is uniformly distributed over a set of size at least  $2^n - q_p$ . Therefore,

$$\Pr[c_{1-b} = x \oplus c_b] \leq \frac{1}{2^n - q_p} \leq \frac{2}{2^n}.$$

Hence, each forkcipher query succeeds with probability at most  $4/2^n$ . Applying the union bound over at most  $q_p$  oracle queries, we obtain

$$\Pr[\mathcal{A}^{\mathbf{F}^\pm} \rightarrow (K, x) \text{ s.t. } \mathbf{C}(K, x) = x] \leq \frac{4q_p}{2^n}.$$

□

**Lemma 8** (Collision resistance of ForkHash). *Let  $\mathbf{F}$  be an ideal forkcipher and  $\mathcal{A}$  be an adversary making at most  $q_p \leq 2^{n-1}$  queries to the forkcipher oracle  $\mathbf{F}^\pm$ . Then,*

$$\mathbf{Adv}_H^{\text{CR}}(\mathcal{A}) \leq \frac{8q_p^2}{2^n},$$

where ForkHash is denoted by  $\mathbf{H}$ .

*Proof.* Let  $\mathcal{A}$  output two distinct tuples  $(N, A, M) \neq (N', A', M')$  such that  $\mathbf{H}(N, A, M) = \mathbf{H}(N', A', M')$ . We bound the probability of this event when  $\mathcal{A}$  makes at most  $q_p$  queries to the underlying ideal forkcipher  $\mathbf{F}^\pm$ . Let

$$\begin{aligned} m_1 \| m_2 \| \cdots \| m_\ell &\leftarrow_n U (= \text{pad}(N, A, M)), \\ m'_1 \| m'_2 \| \cdots \| m'_{\ell'} &\leftarrow_n U' (= \text{pad}(N', A', M')). \end{aligned}$$

We consider the following two cases.

- **Case 1:**  $|A| \neq |A'| \vee |M| \neq |M'|$ : By the definition of  $\text{pad}$ , the final blocks satisfy

$$m_\ell = [A]_{n/2} \| [M]_{n/2} \neq [A']_{n/2} \| [M']_{n/2} = m'_{\ell'}.$$

From the construction of  $\mathbf{H}$  (Algorithm 3), we have

$$\mathbf{H}(N, A, M) = \mathbf{C}(h_\ell, m_\ell) \quad \text{and} \quad \mathbf{H}(N', A', M') = \mathbf{C}(h'_{\ell'}, m'_{\ell'}).$$

Hence, a collision in  $\mathbf{H}$  implies a collision in  $\mathbf{C}$  on two distinct inputs. By Lemma 6, the probability of this event is upper bounded by  $4q_p^2/2^n$ .

- **Case 2:**  $|A| = |A'| \wedge |M| = |M'|$ : In this case,  $\ell = \ell'$  and the final padding blocks are identical, i.e.,  $m_\ell = m'_{\ell'}$ . Let,  $I_i = (h_i, m_i)$  and  $I'_i = (h'_i, m'_i)$  for all  $i = 1, \dots, \ell$ , and let  $I_{\ell+1} = h_{\ell+1}$  and  $I'_{\ell+1} = h'_{\ell+1}$ . Let  $L$  be the *largest* index for which  $I_L \neq I'_L$ . Since  $(N, A, M) \neq (N', A', M')$ , we have  $U \neq U'$ . Hence, there exists an index  $i \in \{1, \dots, \ell\}$  such that  $m_i \neq m'_i$ , ensuring that such an index  $L$  indeed exists.

Now, the hash collision implies that  $h_{\ell+1} = h'_{\ell+1}$ . Therefore, we must have  $L \leq \ell$ . By the maximality of  $L$ , it follows that  $I_{L+1} = I'_{L+1}$ , and in particular,  $h_{L+1} = h'_{L+1}$ . Consequently, we obtain a collision in  $\mathbf{C}$  on two distinct inputs. Thus, by Lemma 6, this event occurs with probability at most  $4q_p^2/2^n$ .

Combining both cases, the result of Lemma 8 follows. □

**Theorem 5.** *Let  $F$  be an ideal forkcipher and  $\mathcal{A}$  be an adversary making at most  $q_p \leq 2^{n-1}$  queries to the forkcipher oracle  $F^\pm$ . Then,*

$$\mathbf{Adv}_{\text{ForkMAC}}^{\text{CR}}(\mathcal{A}) \leq \frac{16q_p^2}{2^n} + \frac{4q_p}{2^n}.$$

*Proof.* Let  $(K, N, A, M) \neq (K', N', A', M')$  be two outputs by the adversary  $\mathcal{A}$ . The adversary  $\mathcal{A}$  will be successful if

$$\text{ForkMAC}(K, N, A, M) = \text{ForkMAC}(K', N', A', M') = \mathsf{T},$$

which implies that  $F^+(K, h, 0) = F^+(K', h', 0) = \mathsf{T}$ , where  $h = H(N, A, M)$  and  $h' = H(N', A', M')$ . Now, we analyze  $\mathcal{A}$ 's success probability in the following cases.

- $(K, h) = (K', h')$ : This holds if and only if  $H(N, A, M) = H(N', A', M')$ , where  $(N, A, M) \neq (N', A', M')$ . By Lemma 8, this occurs with probability at most  $8q_p^2/2^n$ .
- $(K, h) \neq (K', h')$ : Without loss of generality, assume that the query corresponding to  $(K, h, 0)$  occurs after the query corresponding to  $(K', h', 0)$ .

If the last query with  $(K, h)$  is forward, then  $F^+(K, h, 0)$  is uniformly distributed over at least  $2^n - q_p$  unused values. For the adversary to succeed, this output must equal one of the tags produced by the previous forkcipher queries. Since there are at most  $q_p$  such tags, the success probability for any fixed forward query is at most

$$\frac{q_p}{2^n - q_p} \leq \frac{2q_p}{2^n}.$$

Finally, there are at most  $q_p$  forward queries that could be the last query. Applying a union bound over all such queries, the success probability of the adversary is upper bounded by  $2q_p^2/2^n$ .

On the other hand, if the last query with  $(K, h)$  is backward, then  $\mathcal{A}$  succeeds if

$$F^-(K, \mathsf{T}, 0, 0) = H(N, A, M).$$

Let  $z = F^-(K, \mathsf{T}, 0, 0)$ . Since  $H$  outputs the final chaining value of the iterated compression function, there must exist some  $(k, m)$  for which

$$C(k, m) = F^+(k, m, 0) \oplus F^+(k, m, 1) = z.$$

We analyze the advantage for this case in the following cases:

- If the value  $z$  is produced by evaluating  $C$  on the same forkcipher query as  $F^-(K, \mathsf{T}, 0, 0)$ , then  $(k, m) = (K, z)$  and hence  $C(K, z) = z$ . Therefore, the adversary must find a fixed point of  $C$ , which, by Lemma 7, occurs with probability at most  $4q_p/2^n$ .
- If  $z = C(k, m)$  for some previously computed compression value, then there are at most  $q_p$  such values. By the randomness of  $F$ , this happens with probability at most  $2q_p^2/2^n$ .
- Otherwise, the adversary must find  $(k, m)$  such that  $C(k, m) = z$ . By Lemma 5, this occurs with probability at most  $4q_p/2^n$ . Applying a union bound over at most  $q_p$  possibilities gives  $4q_p^2/2^n$ .

Combining all the cases, the result of Theorem 5 follows.  $\square$

**Tightness of the CMT4 security of ForkDTE1 (and ForkDTE2).** We now demonstrate the tightness of the CMT4 bound for ForkDTE1 (and ForkDTE2) established in Theorem 3. The attack proceeds as follows.

1. Choose a key  $K$ , arbitrary values  $N, M \in \{0, 1\}^n$ .
2. Compute  $h_1 = C(0^n, 0^n)$ .
3. Compute  $h_2 = C(h_1, N)$ .
4. Choose  $2^{n/2}$  distinct associated data values  $A_i \in \{0, 1\}^n$ .
5. Compute  $h_3^i = C(h_2, A_i)$ , for all  $i \in [1, 2^{n/2}]$ .
6. Find indices  $a \neq b \in [2^{n/2}]$  such that  $h_3^a = h_3^b$ .
7. Output the two tuples  $(K, N, A_a, M)$  and  $(K, N, A_b, M)$ .

Since the values  $h_3^i$  are independent and uniformly random over  $\{0, 1\}^n$ , we will find such  $a, b$  with a non-negligible probability by the birthday paradox. Note that for such  $A_a, A_b$  we will get a collision for  $H$ , and that will lead to the successful collision pair, as we use the same nonce and message.

*Remark 1.* From the relationship between the CMT1 to CMT4 notions, we can conclude that ForkDTE1 and ForkDTE2 achieve at least the same level of CMT1, CMT2, and CMT3 security as it does CMT4 security. Observe that the above attack is a CMT3 attack on ForkDTE1 and ForkDTE2, so it is a valid CMT4 attack also. But extending this attack to the CMT1 or CMT2 setting appears to be significantly more difficult. In particular, even constructing an attack using two distinct messages seems challenging. The main obstacle is that the same nonce-message pair is used in both the encryption and MAC components, making it difficult to identify inputs that simultaneously satisfy the required collision properties in both components. Therefore, we conjecture that ForkDTE1 and ForkDTE2 may achieve beyond-birthday security against CMT1 and CMT2 adversaries.

## 6 Conclusion

This work settles the committing security landscape of forkcipher-based AEAD schemes. Our analysis shows that, despite earlier negative results, forkcipher-based constructions can indeed achieve strong forms of commitment. In particular, our results suggest that forkciphers provide a suitable primitive for realizing committing security when used within appropriate modes of operation, and highlight the structural features that enable such guarantees.

Taken together, these findings reinforce the viability of forkcipher-based designs for applications requiring strong binding between ciphertexts and their encryption inputs. An interesting direction for future work is to move beyond the ideal forkcipher model and investigate whether comparable guarantees can be achieved under weaker, more standard assumptions. Another interesting direction is to investigate whether ForkDTE1 and ForkDTE2 achieve beyond-birthday CMT1 and CMT2 security, since our CMT3 attack does not readily extend to these settings.

**Acknowledgements.** We would like to thank the anonymous reviewers of CiC 2026 for helpful remarks which improved the paper. Sougata Mandal was partially supported in this work by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation.

## References

- [ABCV98] William Aiello, Mihir Bellare, Giovanni Di Crescenzo, and Ramarathnam Venkatesan. Security amplification by composition: The case of doubly-iterated, ideal ciphers. In Hugo Krawczyk, editor, *Advances in Cryptology - CRYPTO '98, 18th Annual International Cryptology Conference, Santa Barbara, USA, Proceedings*, volume 1462 of *Lecture Notes in Computer Science*, pages 390–407. Springer, 1998. doi:[10.1007/BFB0055743](https://doi.org/10.1007/BFB0055743).
- [ABN18] Michel Abdalla, Mihir Bellare, and Gregory Neven. Robust encryption. *J. Cryptol.*, 31(2):307–350, 2018. doi:[10.1007/S00145-017-9258-8](https://doi.org/10.1007/S00145-017-9258-8).
- [ABV20] Elena Andreeva, Amit Singh Bhati, and Damian Vizár. Nonce-misuse security of the SAEF authenticated encryption mode. In Orr Dunkelman, Michael J. Jacobson Jr., and Colin O’Flynn, editors, *Selected Areas in Cryptography - SAC 2020 - 27th International Conference, Halifax, NS, Canada (Virtual Event), October 21-23, 2020, Revised Selected Papers*, volume 12804 of *Lecture Notes in Computer Science*, pages 512–534. Springer, 2020. doi:[10.1007/978-3-030-81652-0\\_20](https://doi.org/10.1007/978-3-030-81652-0_20).
- [ADG<sup>+</sup>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, *31st USENIX Security Symposium, USENIX Security 2022, Boston, MA, USA, August 10-12, 2022*, pages 3291–3308. USENIX Association, 2022. URL: <https://www.usenix.org/conference/usenixsecurity22/presentation/albertini>.
- [AEG<sup>+</sup>26] Elena Andreeva, Maria Eichlseder, Simon Gerhalter, Marcel Nageler, Oliver Christoph Popa, and Andreas Weninger. Forkcipher-based committing security from explicit collision security assumptions. Cryptology ePrint Archive, Paper 2026/2110, 2026. URL: <https://eprint.iacr.org/2026/2110>.
- [ALP<sup>+</sup>19] Elena Andreeva, Virginie Lallemand, Antoon Purnal, Reza Reyhanitabar, Arnab Roy, and Damian Vizár. Forkcipher: A New Primitive for Authenticated Encryption of Very Short Messages. In Steven D. Galbraith and Shiho Moriai, editors, *ASIACRYPT II*, volume 11922 of *LNCS*, pages 153–182. Springer, 2019. doi:[10.1007/978-3-030-34621-8\\_6](https://doi.org/10.1007/978-3-030-34621-8_6).
- [BAV24] Amit Singh Bhati, Elena Andreeva, and Damian Vizár. OAE-RUP: A strong online AEAD security notion and its application to SAEF. In Clemente Galdi and Duong Hieu Phan, editors, *Security and Cryptography for Networks - 14th International Conference, SCN 2024, Amalfi, Italy, September 11-13, 2024, Proceedings, Part II*, volume 14974 of *Lecture Notes in Computer Science*, pages 117–139. Springer, 2024. doi:[10.1007/978-3-031-71073-5\\_6](https://doi.org/10.1007/978-3-031-71073-5_6).
- [BGP<sup>+</sup>19] Francesco Berti, Chun Guo, Olivier Pereira, Thomas Peters, and François-Xavier Standaert. Strong authenticity with leakage under weak and falsifiable physical assumptions. In Zhe Liu and Moti Yung, editors, *Information Security and Cryptology - 15th International Conference, Inscrypt 2019, Nanjing, China, December 6-8, 2019, Revised Selected Papers*, *Lecture Notes in Computer Science*, pages 517–532. Springer, 2019. doi:[10.1007/978-3-030-42921-8\\_31](https://doi.org/10.1007/978-3-030-42921-8_31).
- [BGPS21] Francesco Berti, Chun Guo, Thomas Peters, and François-Xavier Standaert. Efficient leakage-resilient macs without idealized assumptions. In Mehdi Tibouchi and Huaxiong Wang, editors, *Advances in Cryptology - ASIACRYPT*

- 2021 - 27th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 6-10, 2021, Proceedings, Part II*, volume 13091 of *Lecture Notes in Computer Science*, pages 95–123. Springer, 2021. doi:[10.1007/978-3-030-92075-3\\_4](https://doi.org/10.1007/978-3-030-92075-3_4).
- [BH22] Mihir Bellare and Viet Tung Hoang. Efficient Schemes for Committing Authenticated Encryption. In Orr Dunkelman and Stefan Dziembowski, editors, *EUROCRYPT II*, volume 13276 of *Lecture Notes in Computer Science*, pages 845–875. Springer, 2022. doi:[10.1007/978-3-031-07085-3\\_29](https://doi.org/10.1007/978-3-031-07085-3_29).
- [BKP<sup>+</sup>16] Francesco Berti, François Koeune, Olivier Pereira, Thomas Peters, and François-Xavier Standaert. Leakage-resilient and misuse-resistant authenticated encryption. *IACR Cryptol. ePrint Arch.*, 2016:996, 2016. URL: <http://eprint.iacr.org/2016/996>.
- [Boz25] Andrey Bozhko. Properties of authenticated encryption with associated data (AEAD) algorithms. *RFC*, 9771:1–25, 2025. doi:[10.17487/RFC9771](https://doi.org/10.17487/RFC9771).
- [BPA<sup>+</sup>23] Amit Singh Bhati, Erik Pohle, Aysajan Abidin, Elena Andreeva, and Bart Preneel. Let’s go eevee! A friendly and suitable family of AEAD modes for iot-to-cloud secure computation. In Weizhi Meng, Christian Damsgaard Jensen, Cas Cremers, and Engin Kirda, editors, *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security, CCS 2023, Copenhagen, Denmark, November 26-30, 2023*, pages 2546–2560. ACM, 2023. doi:[10.1145/3576915.3623091](https://doi.org/10.1145/3576915.3623091).
- [BPPS17] Francesco Berti, Olivier Pereira, Thomas Peters, and François-Xavier Standaert. On leakage-resilient authenticated encryption with decryption leakages. *IACR Trans. Symmetric Cryptol.*, 2017(3):271–293, 2017. doi:[10.13154/TOSC.V2017.I3.271-293](https://doi.org/10.13154/TOSC.V2017.I3.271-293).
- [BSL24] Francesco Berti, François-Xavier Standaert, and Itamar Levi. Authenticity in the presence of leakage using a forkcipher. *IACR Commun. Cryptol.*, 1(4):13, 2024. doi:[10.62056/ABKSR-10K](https://doi.org/10.62056/ABKSR-10K).
- [CR22] John Chan and Phillip Rogaway. On committing authenticated-encryption. In Vijayalakshmi Atluri, Roberto Di Pietro, Christian Damsgaard Jensen, and Weizhi Meng, editors, *Computer Security - ESORICS 2022 - 27th European Symposium on Research in Computer Security, Copenhagen, Denmark, September 26-30, 2022, Proceedings, Part II*, volume 13555 of *Lecture Notes in Computer Science*, pages 275–294. Springer, 2022. doi:[10.1007/978-3-031-17146-8\\_14](https://doi.org/10.1007/978-3-031-17146-8_14).
- [DDL24] Nilanjan Datta, Avijit Dutta, Eik List, and Sougata Mandal. FEDT: forkcipher-based leakage-resilient beyond-birthday-secure AE. *IACR Commun. Cryptol.*, 1(2):21, 2024. doi:[10.62056/AGYL86BM](https://doi.org/10.62056/AGYL86BM).
- [DEJ<sup>+</sup>24] Chandranan Dhar, Jordan Ethan, Ravindra Jejurikar, Mustafa Khairallah, Eik List, and Sougata Mandal. Context-committing security of leveled leakage-resilient AEAD. *IACR Trans. Symmetric Cryptol.*, 2024(2):348–370, 2024. doi:[10.46586/TOSC.V2024.I2.348-370](https://doi.org/10.46586/TOSC.V2024.I2.348-370).
- [DGRW18] Yevgeniy Dodis, Paul Grubbs, Thomas Ristenpart, and Joanne Woodage. Fast message franking: From invisible salamanders to encryption. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology - CRYPTO 2018 - 38th Annual International Cryptology Conference*,

- Santa Barbara, CA, USA, August 19-23, 2018, Proceedings, Part I*, volume 10991 of *Lecture Notes in Computer Science*, pages 155–186. Springer, 2018. doi:10.1007/978-3-319-96884-1\\_6.
- [FOR17] Pooya Farshim, Claudio Orlandi, and Razvan Rosie. Security of symmetric primitives under incorrect usage of keys. *IACR Trans. Symmetric Cryptol.*, 2017(1):449–473, 2017. doi:10.13154/TOSC.V2017.I1.449-473.
- [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 - 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 20-24, 2017, Proceedings, Part III*, volume 10403 of *Lecture Notes in Computer Science*, pages 66–97. Springer, 2017. doi:10.1007/978-3-319-63697-9\\_3.
- [KLL20] Hwigeom Kim, Yeongmin Lee, and Jooyoung Lee. Forking tweakable even-mansour ciphers. *IACR Trans. Symmetric Cryptol.*, 2020(4):71–87, 2020. doi:10.46586/TOSC.V2020.I4.71-87.
- [LGR21] Julia Len, Paul Grubbs, and Thomas Ristenpart. Partitioning oracle attacks. In Michael D. Bailey and Rachel Greenstadt, editors, *30th USENIX Security Symposium, USENIX Security 2021, August 11-13, 2021*, pages 195–212. USENIX Association, 2021. URL: <https://www.usenix.org/conference/usenixsecurity21/presentation/len>.
- [Man24] Sougata Mandal. Tweakable forkcipher from ideal block cipher. *IACR Commun. Cryptol.*, 1(3):42, 2024. doi:10.62056/AEY4FBN2HD.
- [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 - 42nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Lyon, France, April 23-27, 2023, Proceedings, Part IV*, volume 14007 of *Lecture Notes in Computer Science*, pages 379–407. Springer, 2023. doi:10.1007/978-3-031-30634-1\\_13.
- [Ost90] Rafail Ostrovsky. Efficient computation on oblivious RAMs. In Harriet Ortiz, editor, *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing, May 13-17, 1990, Baltimore, Maryland, USA*, pages 514–523. ACM, 1990. doi:10.1145/100216.100289.
- [PSV15] Olivier Pereira, François-Xavier Standaert, and Srinivas Vivek. Leakage-resilient authentication and encryption from symmetric cryptographic primitives. In Indrajit Ray, Ninghui Li, and Christopher Kruegel, editors, *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, Denver, CO, USA, October 12-16, 2015*, pages 96–108. ACM, 2015. doi:10.1145/2810103.2813626.
- [RKI24] Mostafizar Rahman, Samir Kundu, and Takanori Isobe. On the key-commitment properties of forkcipher-based aeads. *IACR Commun. Cryptol.*, 1(4):32, 2024. doi:10.62056/AYFHP2FGX.
- [Rog04] Phillip Rogaway. Nonce-based symmetric encryption. In Bimal K. Roy and Willi Meier, editors, *Fast Software Encryption, 11th International Workshop, FSE 2004, Delhi, India, February 5-7, 2004, Revised Papers*, volume 3017 of *Lecture Notes in Computer Science*, pages 348–359. Springer, 2004. doi:10.1007/978-3-540-25937-4\\_22.