On the Security of COMET Authenticated Encryption Scheme

Shay Gueron[1,2], Ashwin Jha[3], and Mridul Nandi[3]
{shay.gueron,ashwin.jha1991,mridul.nandi}@gmail.com

[1]University of Haifa, Israel
[2]Amazon Web Services, U.S.A
[3]Indian Statistical Institute Kolkata, India

Abstract. In this paper, we present a security proof for COMET, the underlying mode of operation of COMET AEAD, a round 1 candidate of the NIST Lightweight Standardization Process. We show that COMET-128 is secure up to 2^{64} bytes of data and 2^{119} offline computations. This validates the security claims of COMET.

Keywords: NIST lightweight cryptography, ICM, COMET, provable security

1 Introduction

Lightweight cryptography has seen a sudden surge in demand due to the recent advancements in the field of Internet of things (IoT). The NIST lightweight cryptography standardization project [1], or NIST LwC project, intends to address this demand by standardizing lightweight authenticated encryption (AE) and cryptographic hash schemes.

The first round of NIST LwC project has 56 candidates, of which around 22 schemes are based on block ciphers. Among these 22 schemes, COMET [2], mixFeed [3], REMUS [4], and TGIF [5] are some of the feedback based schemes, which use nonce and position based re-keying. In this paper we focus on COMET.

COMET is parametrized by the block size of the underlying block cipher. Accordingly, COMET-n means COMET with block size n. It has two versions, one with $n = \kappa$, and the other with $n = \kappa/2$, where n and κ denote the block size and key size of the block cipher. The concrete submissions using COMET mode are based on AES-128/128 [6], Speck-64/128 [7,8], CHAM-128/128 [9], and CHAM-64/128 [9]. Some of the standout features of COMET are as follows:

1. Small State Size: Our main goal is to design AEAD schemes with minimum state size. COMET achieves minimal state size, in the sense that the only state it requires (apart from a constant number of bits) is used for the block cipher, i.e. $(n + \kappa)$-bit state. We believe that this is the smallest possible state size for nonce-based AEAD schemes with security level comparable with COMET.
2. Design Simplicity: The design of COMET is extremely simple. Apart from the block cipher evaluations, it only requires simple shift and XOR operations.
3. Efficiency: This point is closely related to the previous two points. As the design is nonce-based, we are able to keep it single pass, which makes the scheme quite efficient in both hardware and software. Apart from the block cipher call, only 1 shift and at most 2 XOR operations are required per block of AD or PT.

We concentrate on the provable security of the primary version of COMET, i.e. COMET-128. Specifically, our security bound implies that the AE security advantage of COMET-128 is

$$\mathbf{Adv}^{\mathsf{aead}}_{\mathsf{COMET\text{-}128}}(q_e, q_d, \sigma_e, \sigma_d, q_p) \leq \frac{4\sigma_c^2}{2^{256}} + \frac{14\sigma_c q_p}{2^{249}} + \frac{3\sigma_c^2}{2^{128}} + \frac{3.01 q_p}{2^{121}} + \frac{4\sigma_c}{2^{128}} + \frac{q_c}{2^{64}} + \frac{6 q_p \sigma_d}{2^{188.5}},$$

where q_e, q_d, σ_e, σ_d, and q_p, denote the number of encryption queries, number of decryption queries, total number of blocks in all encryption queries, total number of blocks in decryption queries, and the total number of primitive queries, respectively, and $q_c = q_e + q_d$, and $\sigma_c = \sigma_e + \sigma_d$. Note that, $\sigma_c \geq q_c$, and denotes the data complexity, whereas q_p denotes the time complexity.

We summarize the concrete security bounds for different variants of COMET-128 in table 1. A more comprehensive summary is available in the specification file of COMET [2].

Table 1: Summary of security bounds for COMET-128. COMET-128_AES-128/128 and COMET-128_CHAM-128/128 denote the instantiation of COMET-128 by AES-128/128 and CHAM-128/128, respectively.

Submissions	Confidentiality		Integrity	
	Time	Data (in bytes)	Time	Data (in bytes)
COMET-128_AES-128/128	2^{119}	2^{64}	2^{119}	2^{64}
COMET-128_CHAM-128/128	2^{119}	2^{64}	2^{119}	2^{64}

2 Preliminaries

NOTATIONAL SETUP: For a positive integer n, $[n]$ denotes the set $\{1, 2, \ldots, n\}$ and $(n) = \{0, \ldots, n-1\}$. For $n, \kappa \in \mathbb{N}$, $\mathsf{ICPerm}(\kappa, n)$ denotes the set of all families of permutations $\pi_\kappa := \pi(\kappa, \cdot) \in \mathsf{Perm}(n)$, indexed by $\kappa \in \{0,1\}^\kappa$. For non-negative integers n, k, such that $n \geq k$, we define the falling factorial $(n)_k := n!/(n-k)! = n(n-1)\cdots(n-k+1)$. Note that $(n)_k \leq n^k$. Proposition 1 is a well-known inequality result.

Proposition 1. *For all $n, k \in \mathbb{N}$, such that $n \geq k$, we have*

$$\frac{\binom{n}{k}}{n^k} \leq \left(\frac{e}{k}\right)^k.$$

In particular, for $n \geq k \geq 2e$, we have

$$\frac{\binom{n}{k}}{n^k} \leq \frac{1}{2^k}.$$

Proof. We have,

$$\frac{\binom{n}{k}}{n^k} = \frac{n!}{(n-k)!k!n^k} \overset{1}{\leq} \frac{1}{k!} \overset{2}{\leq} \left(\frac{e}{k}\right)^k.$$

where inequality 1 follows from the fact that $(n)_k \leq n^k$, and inequality 2 directly follows from the power series definition of e^k [10], i.e.

$$e^k = \sum_{i=0}^{\infty} \frac{k^i}{i!} \geq \frac{k^k}{k!}.$$

The remaining part follows from a simple substitution of $k = 2e$. □

We fix positive even integers n, r, κ, and t to denote the *block size*, *nonce size*, *key size*, and *tag size*, respectively in bits. We fix $p = \kappa/2$. We use $\{0,1\}^+$ and $\{0,1\}^n$ to denote the set of all non-empty (binary) strings, and n-bit strings, respectively. $\perp$ denotes the empty string and $\{0,1\}^* = \{0,1\}^+ \cup \{\perp\}$. For all practical purposes: we use little-endian format of indexing, and assume all binary strings are *byte-oriented*, i.e. belong in $(\{0,1\}^8)^*$. For any string $B \in \{0,1\}^+$, $|B|$ denotes the number of bits in B, and for $0 \leq i \leq |B|-1$, b_i denotes the i-th bit of B, i.e. $B = b_{|B|-1} \cdots b_0$. For $B \in \{0,1\}^+$, $(B_{\ell-1}, \ldots, B_0) \xleftarrow{n} B$, denotes the n-bit *block parsing* of B into $(B_{\ell-1}, \ldots, B_0)$, where $|B_i| = n$ for $0 \leq i \leq \ell-2$, and $1 \leq |B_{\ell-1}| \leq n$. For $A, B \in \{0,1\}^+$, and $|A| = |B|$, $A \oplus B$ denotes the "bitwise XOR" operation on A and B. For $A, B \in \{0,1\}^+$, $A \| B$ denotes the "string concatenation" operation on A and B. For any $B \in \{0,1\}^+$ and a non-negative integer s, $B \ll s$ and $B \lll s$ denote the "left shift by s" and "circular left shift by s" operations on B, respectively. The notations for right shift and circular right shift are analogously defined using $\gg$ and $\ggg$, respectively.

The set $\{0,1\}^p$ can be viewed as the finite field $\mathbb{F}_2^p$ consisting of 2^p elements. We interchangeably think of an element $B \in \mathbb{F}_2^p$ in any of the following ways: (i) as a p-bit string $b_{p-1} \ldots b_1 b_0 \in \{0,1\}^p$; (ii) as a polynomial $B(x) = b_{p-1}x^{p-1} + b_{p-2}x^{p-2} + \cdots + b_1 x + b_0$ over the field $\mathbb{F}_2$; (iii) a non-negative integer $b < 2^p$; (iv) an abstract element in the field. Addition in $\mathbb{F}_2^p$ is just bitwise XOR of two p-bit strings, and hence denoted by $\oplus$. $P(x)$ denotes the primitive polynomial used to represent the field $\mathbb{F}_2^p$, and α denotes a fixed primitive element in this representation. The multiplication of $A, B \in \mathbb{F}_2^p$ is defined as $A \odot B := A(x) \cdot B(x) \pmod{P(x)}$, i.e. polynomial multiplication modulo $P(x)$ in $\mathbb{F}_2$. For any $B \in \mathbb{F}_2^p$, multiplication with α is computationally efficient. We demonstrate this for $p = 64$, as we will fix $\kappa = 128$ in our submissions. For $p = 64$, $P(x) = x^{64} + x^4 + x^3 + x + 1$ is a primitive polynomial, and we let α to denote the primitive element $2 \in \mathbb{F}_2^{64}$. Then for any $B \in \mathbb{F}_2^{64}$, we have

$$A \odot \alpha = \begin{cases} A \ll 1 & \text{if } a_{|A|-1} = 0, \\ (A \ll 1) \oplus 0^{59}11011 & \text{if } a_{|A|-1} = 1. \end{cases}$$

For $q \in \mathbb{N}$, X^q denotes the q-tuple $(X_1, X_2, \ldots, X_q)$. For $q \in \mathbb{N}$, for any set $\mathcal{X}$, $(\mathcal{X})_q$ denotes the set of all q-tuples with distinct elements from $\mathcal{X}$. We use short hand notation $\exists^*$ to represent the phrase "there exists distinct". For a finite set $\mathcal{X}$, $\mathsf{X} \xleftarrow{\$} \mathcal{X}$ denotes the uniform and random sampling of X from $\mathcal{X}$.

2.1 Authenticated Encryption: Definition and Security Model

Authentication Encryption with Associated Data: An authenticated encryption scheme with associated data functionality, or AEAD in short, is a tuple of algorithms $\mathsf{AE} = (\mathsf{E}, \mathsf{D})$, defined over the *key space* $\mathcal{K}$, *nonce space* $\mathcal{N}$, *associated data space* $\mathcal{A}$, *message space* $\mathcal{M}$, *ciphertext space* $\mathcal{C}$, and *tag space* $\mathcal{T}$, where:

$$\mathsf{E} : \mathcal{K} \times \mathcal{N} \times \mathcal{A} \times \mathcal{M} \to \mathcal{C} \times \mathcal{T} \quad \text{and} \quad \mathsf{D} : \mathcal{K} \times \mathcal{N} \times \mathcal{A} \times \mathcal{C} \times \mathcal{T} \to \mathcal{M} \cup \{\perp\}.$$

Here, Eand Dare called the encryption and decryption algorithms, respectively, of AE. Further, it is required that $\mathsf{D}(K, N, A, \mathsf{E}(K, N, A, M)) = M$ for any $(K, N, A, M) \in \mathcal{K} \times \mathcal{N} \times \mathcal{A} \times \mathcal{M}$. For all key $K \in \mathcal{K}$, we write $\mathsf{E}_K(\cdot)$ and $\mathsf{D}_K(\cdot)$ to denote $\mathsf{E}(K, \cdot)$ and $\mathsf{D}(K, \cdot)$, respectively. In this paper, we have $\mathcal{K}, \mathcal{N}, \mathcal{A}, \mathcal{M}, \mathcal{T} \subseteq \{0,1\}^+$ and $\mathcal{C} = \mathcal{M}$, so we use $\mathcal{M}$ instead of $\mathcal{C}$ wherever necessary.

AEAD Security in the Ideal Cipher Model: A block cipher with key size κ and block size n is a family of permutations $\mathsf{IC} \in \mathsf{ICPerm}(\kappa, n)$. For $K \in \{0,1\}^\kappa$, we denote $\mathsf{IC}_K(\cdot) = \mathsf{IC}^+_K(\cdot) := \mathsf{IC}(K, \cdot)$, and $\mathsf{IC}^-_K(\cdot) := \mathsf{IC}^{-1}(K, \cdot)$. *A block cipher is said to be an ideal cipher if for all* $K \in \{0,1\}^\kappa$, $\mathsf{IC}_K \xleftarrow{\$} \mathsf{Perm}(n)$.

Let Func denote the set of all functions from $\mathcal{N} \times \mathcal{A} \times \mathcal{M}$ to $\mathcal{M} \times \mathcal{T}$, and $\Gamma \xleftarrow{\$} \mathsf{Func}$. Let $\perp$ denote the degenerate function from $(\mathcal{N}, \mathcal{A}, \mathcal{M}, \mathcal{T})$ to $\{\perp\}$. For brevity, we denote the oracle corresponding to a function (like E, IC etc.) by that function itself. A bidirectional access to IC is denoted by the superscript $\pm$.

Definition 1. *Let* $\mathsf{AE}_{\mathsf{IC}}$ *be an AEAD scheme, based on the ideal cipher* IC*, defined over* $(\mathcal{K}, \mathcal{N}, \mathcal{A}, \mathcal{M}, \mathcal{T})$*. The AEAD advantage of an adversary* $\mathscr{A}$ *against* $\mathsf{AE}_{\mathsf{IC}}$ *is defined as,*

$$\mathbf{Adv}^{\mathsf{aead}}_{\mathsf{AE}_{\mathsf{IC}}}(\mathscr{A}) := \left| \Pr_{\substack{\mathsf{K} \xleftarrow{\$} \mathcal{K} \\ \mathsf{IC}^\pm}} \left[\mathscr{A}^{\mathsf{E}_\mathsf{K}, \mathsf{D}_\mathsf{K}, \mathsf{IC}^\pm} = 1 \right] - \Pr_{\Gamma, \mathsf{IC}^\pm} \left[\mathscr{A}^{\Gamma, \perp, \mathsf{IC}^\pm} = 1 \right] \right|. \tag{1}$$

Here $\mathscr{A}^{\mathsf{E}_\mathsf{K}, \mathsf{D}_\mathsf{K}, \mathsf{IC}^\pm}$ denotes $\mathscr{A}$'s response after its interaction with E_K, D_K, and $\mathsf{IC}^\pm$, respectively. Similarly, $\mathscr{A}^{\Gamma, \perp, \mathsf{IC}^\pm}$ denotes $\mathscr{A}$'s response after its interaction with Γ, $\perp$, and $\mathsf{IC}^\pm$.

In this paper, we assume that the adversary is *non-trivial* and *nonce respecting*, i.e. it never makes a duplicate query, it never makes a query for which the response is already known due to some previous query, and it does not repeat nonce values in encryption queries. Throughout, we use the following notations to parametrize the adversary's resources:

- q_e and q_d denote the number of queries to E_K and D_K, respectively. σ_e and σ_d denote the sum of input (associated data and message) lengths across all encryption and decryption, respectively, queries. We sometime also write $q_c = q_e + q_d$ and $\sigma_c = \sigma_e + \sigma_d$ to denote the combined construction query resources.
- q_+ and q_- denote the number of queries to IC^+ and IC^-, respectively. We sometime also use $q_p = q_+ + q_-$, to denote the combined primitive query resources.

Any adversary $\mathscr{A}$ that abides by the above given resources is referred as a $(q_e, q_d, \sigma_e, \sigma_d, q_+, q_-)$-adversary. We remark here that q_c and σ_c correspond to the *online or data complexity*, and q_p corresponds to the *offline or time complexity* of the adversary.

3 The COMET Mode of Operation

COunter Mode Encryption with authentication Tag, or COMET in abbreviation, is a block cipher mode of operation that provides authenticated encryption with associated data (henceforth "AEAD") functionality. At a very high level, it can be viewed as a mixture of CTR [11], Beetle [12,13], and COFB [14,15] modes of operation. In this section we provide complete specification of the COMET family of AEAD ciphers.

3.1 Parameters

COMET is primarily parameterized by the block size n of the underlying block cipher, where $n \in \{64, 128\}$. In other words we allow block ciphers with 64-bit and 128-bit block sizes. We simply write COMET-n to denote COMET with the particular choice of n, and skip the parametrization whenever the context applies to both variants. The secondary parameters are set according to the value of n in the following manner.

- COMET-128: In this version $n = 128$, $r = 128$, $\kappa = 128$, $t = 128$, and $p = 64$.
- COMET-64: In this version $n = 64$, $r = 120$, $\kappa = 128$, $t = 64$, and $p = 64$.

In both variants, we use the primitive polynomial $P(x) = x^{64} + x^4 + x^3 + x + 1$ to represent the field $\mathbb{F}_2^{64} = \{0, 1\}^{64}$, and fix the primitive element $\alpha = 2$.

3.2 Description of COMET

Algorithms 1-3 give the complete algorithmic description of the mode, and figure 1 illustrates the major components of the encryption/decryption process. In the remainder of this subsection, we give a high level description of the main modules (given in algorithm 2) used in the encryption/decryption (described in algorithm 1) process.

- init: Apart from some book-keeping operations, the major task of this module is to create the initial state using the public nonce N and the secret key K. This initial state derivation is the only stage where the two versions of COMET, namely, COMET-128 and COMET-64 vary. The state can be viewed as an $(n+\kappa)$-bit concatenated string $Y\|Z$ made up of n-bit string Y (also called the Y-state) and κ-bit string Z (also called Z-state). In this notation the initial state is $(Y_0, Z_0) = Y_0\|Z_0$. In case of COMET-128, we define $Y_0 = K$ and $Z_0 = E_K(N)$, as described in "**function** init_state_128" of algorithm 3. In case of COMET-64, we use $Y_0 = E_K(0)$ and $Z_0 = K \oplus 0^8\|N$, as described in "**function** init_state_64" of algorithm 3.
- proc_ad: This module is responsible for the associated data (AD) processing. At the start of the processing a control bit indicating start of non-empty AD is XORed to the 5^{th} most significant bit (msb) of the current Z-state. The AD data is absorbed, n bits at a time, using "**function** round" of algorithm 2. In case of partial last block a control bit indicating partial block is XORed to the 4^{th} msb of the Z-state before the processing of the last block.
- proc_pt: This module is responsible for the plaintext (PT) processing. At the start of the processing a control bit indicating start of non-empty PT is XORed to the 3^{rd} msb of the current Z-state. PT processing is similar to AD processing except for the fact that we squeeze out n-bit ciphertext as well. In case of partial last block a control bit indicating partial block is XORed to the 2^{nd} msb of the Z-state before the processing of the last block.
- proc_ct: This module is responsible for ciphertext (CT) processing. It is symmetrical to proc_pt.
- proc_tg: This module is responsible for tag generation. Before the tag generation a control bit indicating the tag generation call is XORed to the msb of the current Z-state.

Algorithm 1 Encryption/Decryption algorithm in COMET.

function COMET-$n_{[E]}$.E(K, N, A, M)
  $C \leftarrow \perp$
  $(Y_0, Z_0, a, m, \ell) \leftarrow$ init(K, N, A, M)
  if $a \neq 0$ **then**
    $(Y_a, Z_a) \leftarrow$ proc_ad(Y_0, Z_0, A)
  if $m \neq 0$ **then**
    $(Y_\ell, Z_\ell, C) \leftarrow$ proc_pt(Y_a, Z_a, M)
  $T \leftarrow$ proc_tg(Y_ℓ, Z_ℓ)
  return (C, T)

function COMET-$n_{[E]}$.D(K, N, A, C, T)
  $M \leftarrow \perp$
  is_auth $\leftarrow 0$
  $(Y_0, Z_0, a, m, \ell) \leftarrow$ init(K, N, A, C)
  if $a \neq 0$ **then**
    $(Y_a, Z_a) \leftarrow$ proc_ad(Y_0, Z_0, A)
  if $m \neq 0$ **then**
    $(Y_\ell, Z_\ell, M) \leftarrow$ proc_ct(Y_a, Z_a, C)
  $T' \leftarrow$ proc_tg(Y_ℓ, Z_ℓ)
  if $T' = T$ **then**
    is_auth $\leftarrow 1$
  else
    $M \leftarrow \perp$
  return (is_auth, M)

Algorithm 2 Main modules of COMET.

```
function init(K, N, A, M)
    if n = 64 then
        (Y_0, Z_0) ← init_state_64(K, N)
    else
        (Y_0, Z_0) ← init_state_128(K, N)
    a ← ⌈|A|/n⌉
    m ← ⌈|M|/n⌉
    ℓ ← a + m
    return (Y_0, Z_0, a, m, ℓ)

function round(Y', Z', I, b)
    Z ← get_blk_key(Z')
    X ← IC(Z, Y')
    if b = 0 then
        Y ← update(X, I, 0)
        return (Y, Z)
    else
        (Y, O) ← update(X, I, b)
        return (Y, Z, O)

function proc_ad(Y_0, Z_0, A)
    (A_{a-1}, ..., A_0) ← parse(A)
    Z_0 ← Z_0 ⊕ 000010^{κ-5}
    for i = 0 to a − 2 do
        (Y_{i+1}, Z_{i+1}) ← round(Y_i, Z_i, A_i, 0)
    if n ∤ |A_{a-1}| then
        Z_{a-1} ← Z_{a-1} ⊕ 000100^{κ-5}
    (Y_a, Z_a) ← round(Y_{a-1}, Z_{a-1}, A_{a-1}, 0)
    return (Y_a, Z_a)

function proc_pt(Y_a, Z_a, M)
    (M_{m-1}, ..., M_0) ← parse(M)
    Z_a ← Z_a ⊕ 001000^{κ-5}
    for j = 0 to m − 2 do
        k ← a + j
        (Y_{k+1}, Z_{k+1}, C_j) ← round(Y_k, Z_k, M_j, 1)
    if n ∤ |M_{m-1}| then
        Z_{ℓ-1} ← Z_{ℓ-1} ⊕ 010000^{κ-5}
    (Y_ℓ, Z_ℓ, C_{m-1}) ← round(Y_{ℓ-1}, Z_{ℓ-1}, M_{m-1}, 1)
    C ← (C_{m-1}, ..., C_0)
    return (Y_ℓ, Z_ℓ, C)

function proc_ct(Y_a, Z_a, C)
    (C_{m-1}, ..., C_0) ← parse(C)
    Z_a ← Z_a ⊕ 001000^{κ-5}
    for j = 0 to m − 2 do
        k ← a + j
        (Y_{k+1}, Z_{k+1}, M_j) ← round(Y_k, Z_k, C_j, 2)
    if n ∤ |C_{m-1}| then
        Z_{ℓ-1} ← Z_{ℓ-1} ⊕ 010000^{κ-5}
    (Y_ℓ, Z_ℓ, M_{m-1}) ← round(Y_{ℓ-1}, Z_{ℓ-1}, C_{m-1}, 2)
    M ← (M_{m-1}, ..., M_0)
    return (Y_ℓ, Z_ℓ, M)

function proc_tg(Y_ℓ, Z_ℓ)
    Z_ℓ ← Z_ℓ ⊕ 100000^{κ-5}
    Z_{ℓ+1} ← get_blk_key(Z_ℓ)
    T ← IC(Z_{ℓ+1}, Y_ℓ)
    return T
```

4 Security of COMET

We give the combined AE security of COMET-128 in the ideal cipher model, as explained in section 2.1. Specifically, in theorem 1, we bound the AE advantage of any adversary against COMET-n initialized with an ideal cipher IC, where $n = \kappa$.

Theorem 1. *For $n \geq 16$, $\sigma_e, \sigma_d < 2^{n-1}$, $q_p < 2^{\kappa-1}$, and $(q_e, q_d, \sigma_e, \sigma_d, q_+, q_-)$-adversary $\mathscr{A}$ we have*

$$\mathbf{Adv}^{\mathsf{aead}}_{\mathsf{COMET}}(\mathscr{A}) \leq \frac{3\sigma_e\sigma_d + 3nq_p}{2^\kappa} + \frac{q_e + q_d}{2^{n/2}} + \frac{2\sigma_e^2 + 2\sigma_c q_p + 6nq_pq_d + 4q_pq_d + 2\sigma_d\sigma_e}{2^{\kappa+n}} + \frac{6\sqrt{n}q_p\sigma_d}{2^{\kappa+n/2}} + \frac{8 + 2q_p + 4\sigma_e}{2^n}.$$

The proof of theorem 1 is given in the rest of this section. First, we develop some notations to suite our main proof tool, the so-called coefficients H technique [16,17,18].

Let $\mathscr{A}$ be a computationally[1] unbounded adversary, whence it is deterministic. The adversary $\mathscr{A}$'s goal is to distinguish between the real oracle, $\mathcal{O}_1 = (\mathsf{COMET.E_K}, \mathsf{COMET.D_K}, \mathsf{IC}^{\pm})$, and the ideal oracle, $\mathcal{O}_0 = (\Gamma, \perp, \mathsf{IC}^{\pm})$. We denote the query-response tuple of $\mathscr{A}$'s interaction with its oracle by a transcript $\omega = \{\omega_e, \omega_d, \omega_p\}$, where $\omega_e := \{(N^i, A^i, M^i, C^i, T^i)_{i\in[q_e]}\}$, $\omega_d := \{(\bar{N}^j, \bar{A}^j, \bar{C}^j, \bar{T}^j, \bar{D}^j)_{j\in[q_e]}\}$, and $\omega_p := \{(\widehat{Z}^k, \widehat{Y}^k, \widehat{X}^k)_{k\in[q_p]}\}$. Here,

- $(N^i, A^i, M^i, C^i, T^i)$ denotes the i-th encryption query-response tuple, where N^i, A^i, M^i, C^i, and T^i, denote the nonce, associated data, message, ciphertext, and tag, respectively. Let $|A^i|/n = a^i$, $|C^i|/n = |M^i|/n = m^i$, and $\ell^i = a^i + m^i$.
- $(\bar{N}^j, \bar{A}^j, \bar{C}^j, \bar{T}^j, \bar{D}^j)$ denotes the j-th decryption query-response tuple, where $\bar{N}^j$, $\bar{A}^j$, $\bar{C}^j$, $\bar{T}^j$, and $\bar{D}^j$, denote the nonce, associated data, ciphertext, tag, and the authentication result, respectively. $\bar{D}^j$ equals

[1] This is the total computational time that the adversary spents (which includes the time taken for querying the AE scheme or the ideal cipher).

Algorithm 3 Various sub-modules of COMET.

```
function chop(I, ℓ)
    if ℓ > n then
        return ⊥
    else
        return i_{ℓ−1} ... i_0

function parse(I)
    ℓ = ⌈|I|/n⌉
    if ℓ = 0 then
        return ⊥
    else
        (I_{ℓ−1}, ..., I_0) ←^n I
        return (I_{ℓ−1}, ..., I_0)

function opt_pad0*1(I)
    if |I| = 0 or n ∤ |I| then
        ξ = n − (|I| mod n)
        I ← 0^{ξ−1}1‖I
    return I

function permute(Z')
    (Z'_1, Z'_0) ←^p Z'
    Z_0 ← Z'_0 ⊙ α
    Z ← (Z'_1, Z_0)
    return Z

function init_state_128(K, N)
    Y ← K
    Z ← IC(K, N)
    return (Y, Z)

function init_state_64(K, N)
    Y ← IC(K, 0)
    Z ← K ⊕ 0^{κ−r}‖N
    return (Y, Z)

function shuffle(X')
    (X'_3, X'_2, X'_1, X'_0) ←^{n/4} X'
    X_2 ← X'_2 ⋙ 1
    X ← (X'_1, X'_0, X_2, X'_3)
    return X

function get_blk_key(Z')
    Z ← permute(Z')
    return Z

function update(X, I, b)
    if b = 0 then
        Y ← X ⊕ opt_pad0*1(I)
        return Y
    else
        X' ← shuffle(X)
        O ← chop(X', |I|) ⊕ I
        if b = 1 then
            Y ← X ⊕ opt_pad0*1(I)
        else if b = 2 then
            Y ← X ⊕ opt_pad0*1(O)
        return (Y, O)
```

to a message $\bar{M}^j$ when authentication succeeds, and $\perp$ otherwise. Let $|\bar{A}^j|/n = \bar{a}^j$ and $|\bar{C}^j|/n = \bar{m}^j$, and $\bar{\ell}^j = \bar{a}^j + \bar{m}^j$.

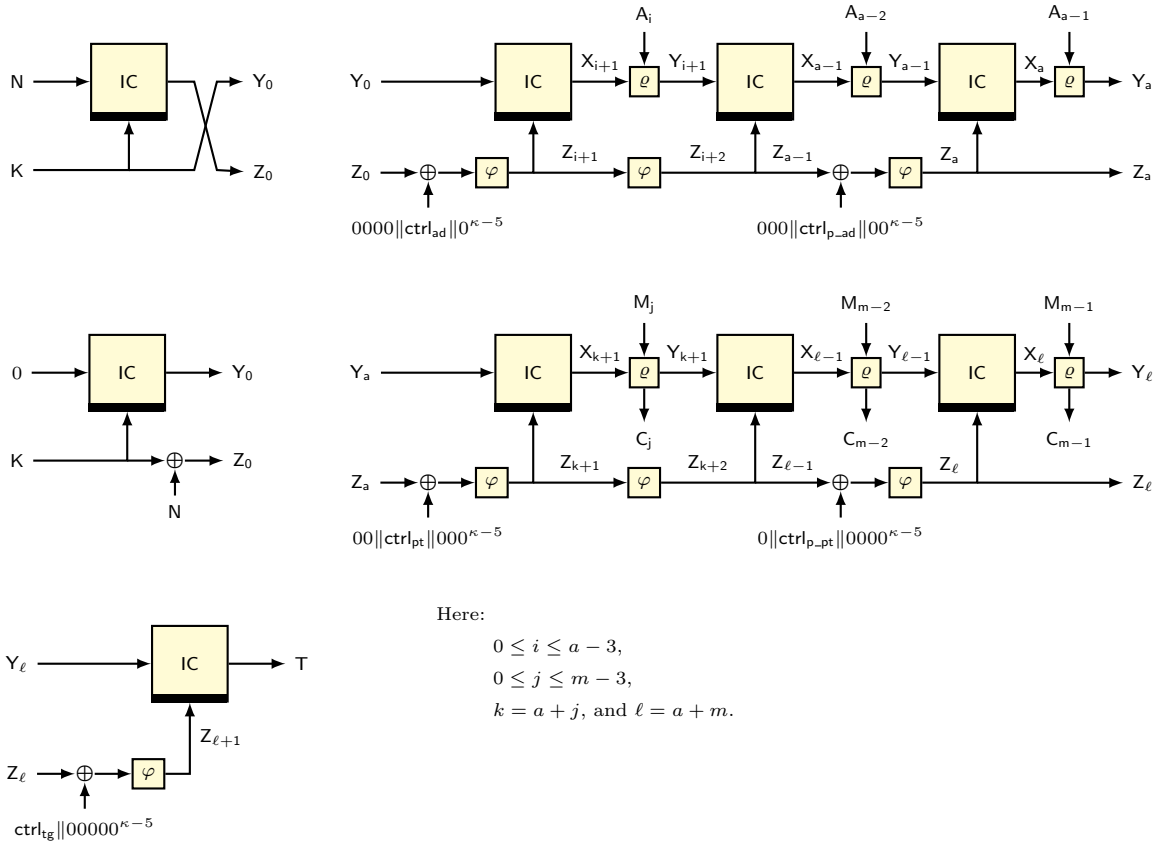
– $(\widehat{Z}^k, \widehat{Y}^k, \widehat{X}^k, \widehat{d}^k)$ denotes the k-th primitive query-response tuple, where $\widehat{Z}^k$, $\widehat{Y}^k$, $\widehat{X}^k$, and $\widehat{d}^k$, denote the key, input, output, and direction of query, respectively. $\widehat{d}^k = +$ if the k-th query is forward, and $\widehat{d}^k = -$ if the k-th query is backward.

In addition, we modify the distinguishing game a little bit, where the oracles release extra information to the adversary after the query-response phase, but before it outputs its decision bit. This extra information is the master key K, and internal variables X (outputs of the block cipher) generated during the encryption query processing, which are defined analogous to figure 1, and algorithm 1-3. Note that, the release of these variables does not decrease the distinguishing advantage.

Real Oracle Description: We first describe the real oracle. The real oracle has access to $\mathsf{IC}^{\pm}$. It faithfully responds to $\mathscr{A}$'s encryption, decryption, and primitive queries, using $\mathsf{IC}^{\pm}$, and after the query phase is over it releases the additional information.

Ideal Oracle Description: The ideal oracle works as follows:

– For the i-the encryption query:
  • Sample $X^i_1, X^i_2, \ldots, X^i_\ell, T^i \leftarrow_{\$} \{0,1\}^n$, sets $C^i_j = \mathsf{shuffle}(X^i_{a+j+1}) \oplus M^i_j$ for all $j \in (m^i]$, and finally returns $C^i_0, \ldots, C^i_{m^i-1}, T^i$.
  • Sets $Y^i_j = X^i_j \oplus A^i_j$ for $j \in [a^i]$, and $Y^i_j = X^i_j \oplus M^i_j$ for $j \in \{a^i+1, \ldots, a^i+m^i\}$.
– For the i-th decryption query: the ideal oracle always returns $\perp$.
– For the i-th primitive query: if $\widehat{d}^i = +$, then it returns $\widehat{X}^i = \mathsf{IC}^+_{\widehat{Z}^i}(\widehat{Y}^i)$, otherwise it returns $\widehat{Y}^i = \mathsf{IC}^-_{\widehat{Z}^i}(\widehat{X}^i)$.
– After the query-response phase is over, the ideal oracle returns $\mathsf{K} \leftarrow_{\$} \{0,1\}^\kappa$, and $(X^i)_{i\in[q_e]}$. It also returns $(Z^i_0, Y^i_0) = \mathsf{init}(K, N^i, A^i, M^i)$ for all $i \in [q_e]$.



Here:

$0 \le i \le a-3$,
$0 \le j \le m-3$,
$k = a + j$, and $\ell = a + m$.

Fig. 1: Schematic diagram of different modules used in the encryption algorithm of COMET for non empty AD and PT. From top to bottom and left to right, we have the following modules: init_state_128, init_state_64, proc_tg, proc_ad, and proc_pt. φ and ϱ denote the functional view of sub-modules permute and shuffle from algorithm 3, respectively. See algorithm 1-3 for more details.

Now, consider a decryption query $i \in [q_d]$. If $\bar{N}^i \neq N^j$, for all $j \in [q_e]$, then we define the index of longest common prefix, denoted δ_i as -1. If there exists a unique index $j \in [q_e]$, such that $\bar{N}^i = N^j$, then we have

$$\delta_i := \begin{cases} \max_{\bar{C}^i_{0\ldots k-1} = C^j_{0\ldots k-1}} (\bar{a}^i + k) & \text{if } \bar{A}^i = A^j \wedge (\bar{A}^i, \bar{C}^i) \neq (A^j, C^j) \\ \max_{\bar{A}^i_{0\ldots k-1} = A^j_{0\ldots k-1}} (k) & \text{otherwise.} \end{cases}$$

It is clear that whenever $\delta_i \geq 0$, then $(\bar{Z}^i_0, \bar{Y}^i_0) = (Z^j_0, Y^j_0)$. Further, $\bar{X}^i_k$, and $\bar{Y}^i_k$ are determined for all $k \in [\delta_i + 1]$, due to X^j_k, Y^j_k, and $\bar{C}^i_{\delta_i}$.

We denote by Θ_1 and Θ_0, the random transcript variable when $\mathscr{A}$ interacts with $\mathcal{O}_1$ (the real oracle) and $\mathcal{O}_0$ (the ideal oracle). A transcript ω is said to be *attainable* if $\Pr[\Theta_0 = \omega] > 0$. Let Ω denote the set of all attainable transcripts.

Main Result of Coefficients H Technique: The Coefficients H Technique works in two steps. First, it identifies a set of bad transcripts, denoted $\Omega_{\mathsf{bad}} \subset \Omega$. Second, suppose for some $\epsilon \geq 0$, and for all $\omega \in \Omega \setminus \Omega_{\mathsf{bad}}$, we have

$$\frac{\Pr[\Theta_1 = \omega]}{\Pr[\Theta_0 = \omega]} \geq 1 - \epsilon.$$

Then the main result of coefficients H technique [16,17] states that,

$$\mathbf{Adv}^{\mathsf{aead}}_{\mathsf{COMET}}(\mathscr{A}) \leq \epsilon + \Pr[\Theta_0 \in \Omega_{\mathsf{bad}}]. \tag{2}$$

Note that, while computing the probability of realizing a transcript, we do not consider the random coin of $\mathscr{A}$. This is due to the fact that $\mathscr{A}$ is deterministic.

A proof for Eq. (2) result is readily available in literature including [17,18,19]. So, to apply coefficients H technique we have to compute a lower bound on the ratio of real to ideal world probabilities of realizing good transcript, i.e. $1-\epsilon$, and upper bound the probability of realizing a bad transcript in ideal world, i.e. $\Pr[\Theta_0 \in \Omega_{\mathsf{bad}}]$.

4.1 Bad Transcripts and Their Analysis

Now, we describe the set of bad transcripts, and then bound the probability of realizing such transcript in the ideal world. We first need some more definitions to define certain special bad events.

We write $\mathsf{domain}(\omega_p) = \{(\widehat{Z}_1, \widehat{Y}_1), \ldots, (\widehat{Z}_{q_p}, \widehat{Y}_{q_p})\}$ and $\mathsf{range}(\omega_p) = \{(\widehat{Z}_1, \widehat{X}_1), \ldots, (\widehat{Z}_{q_p}, \widehat{X}_{q_p})\}$. For all $i \in [q_d]$, we write δ_i' to denote the largest block index such that $\bar{X}^i_{\delta_i+1}, ..., \bar{X}^i_{\delta_i'}$ is in $\mathsf{range}(\omega_p)$. If $\bar{X}^i_{\delta_i+1}$ is not in $\mathsf{domain}(\omega_p)$, then $\delta_i' = \delta_i$.

Labeled Walk: Given the primitive query list ω_p, we define a labeled directed graph $\mathcal{G}_{\omega_p}$ over the set of vertices $\mathsf{domain}(\omega_p)$ as follows: A directed edge $(\widehat{Z}_i, \widehat{Y}_i) \to (\widehat{Z}_j, \widehat{Y}_j)$ with label C (also denoted $(\widehat{Z}_i, \widehat{Y}_i) \xrightarrow{C} (\widehat{Z}_j, \widehat{Y}_j)$) is in the graph if $\widehat{Z}_j = \widehat{Z}_i \odot \alpha$ and $\mathsf{shuffle}(\widehat{X}_i) \oplus \widehat{X}_i \oplus C = \widehat{Y}_j$. This can be similarly extended to a labeled walk $\mathcal{W}$ from a node W_0 to W_k as

$$\mathcal{W} : W_0 \xrightarrow{C_1} W_1 \xrightarrow{C_2} \cdots \xrightarrow{C_k} W_k.$$

We simply denote it as $W_0 \xrightarrow{C} W_k$, where $C = (C_1, \ldots, C_k)$. Here k is the length of the walk.

Definition 2. *We say that a set of labeled walks $\mathcal{C}_{C,T} := \{\mathcal{W}_1, \ldots, \mathcal{W}_t\}$ forms a* multi-chain with a label $(C := (C_1, \ldots, C_k), T)$ *in the graph $\mathcal{G}_{\omega_p}$ if for all $1 \le i \le t$, $\mathcal{W}_i : (\hat{Z}^i_0, \hat{Y}^i_0) \xrightarrow{C} (\hat{Z}^i_k, \hat{Y}^i_k)$ and $\hat{Y}^1_0 = \hat{Y}^2_0 = \cdots = \hat{Y}^t_0$ and $\hat{X}^1_{k+1} = \hat{X}^2_{k+1} = \cdots = \hat{X}^t_{k+1} = T$. We also call $\mathcal{C}_C$ a multi-chain of length k, and t the size of $\mathcal{C}_C$.*

Maximum size of the set of multi-chain of length k (with some label c) is denoted as Λ_k (which is induced by ω_p).

Bad Transcripts: The ideal world transcript is said to be bad if one of the following conditions occur:

- $\mathtt{B1}$: $\exists i \in [q_e]$, $j \in [m^i]$, such that $\mathsf{Z}^i_j = \mathsf{K}$.
- $\mathtt{B2}$: $\exists i \in [q_d]$, $j \in [\bar{m}^i]$, such that $\bar{\mathsf{Z}}^i_j = \mathsf{K}$.
- $\mathtt{B3}$: $\exists i \in [q_p]$, such that $\widehat{\mathsf{Z}}^i = \mathsf{K}$.
- $\mathtt{B4}$: $\exists i \in [q_e]$, such that $\mathsf{Z}^i_0 = *\|0^{n/2}$.
- $\mathtt{B5}$: $\exists i \in [q_d]$, such that $\bar{\mathsf{Z}}^i_0 = *\|0^{n/2}$.
- $\mathtt{B6}$: $\exists (i,j) \in [q_e] \times [m^i], (i',j') \in [q_d] \times [\bar{m}^{i'}]$, such that $\mathsf{N}^i \neq \bar{\mathsf{N}}^{i'}$ and $\mathsf{Z}^i_j = \bar{\mathsf{Z}}^{i'}_{j'}$.
- $\mathtt{B7}$: $\exists (i,j) \in [q_e] \times [m^i], (i',j') \in [q_e] \times [m^{i'}]$, such that $(\mathsf{Z}^i_j, \mathsf{Y}^i_j) = (\mathsf{Z}^{i'}_{j'}, \mathsf{Y}^{i'}_{j'})$.
- $\mathtt{B8}$: $\exists (i,j) \in [q_e] \times [m^i], (i',j') \in [q_e] \times [m^{i'}]$, such that $(\mathsf{Z}^i_j, \mathsf{X}^i_j) = (\mathsf{Z}^{i'}_{j'}, \mathsf{X}^{i'}_{j'})$.
- $\mathtt{B9}$: $\exists (i,j) \in [q_e] \times [m^i]$ and $i' \in [q_p]$, such that $(\mathsf{Z}^i_j, \mathsf{Y}^i_j) = (\widehat{\mathsf{Z}}^{i'}, \widehat{\mathsf{Y}}^{i'})$.
- $\mathtt{B10}$: $\exists (i,j) \in [q_e] \times [m^i]$ and $i' \in [q_p]$, such that $(\mathsf{Z}^i_j, \mathsf{X}^i_j) = (\widehat{\mathsf{Z}}^{i'}, \widehat{\mathsf{X}}^{i'})$.
- $\mathtt{B11}$: $\exists i \in [q_d]$ such that $\delta_i \ge 0$, $\delta_i' = \bar{\ell}^i$ and $\bar{\mathsf{X}}^i_{\bar{\ell}^i+1} = \bar{\mathsf{T}}^i$.
- $\mathtt{B12}$: $\exists i \in [q_d], (i',j') \in [q_e] \times [m^{i'}]$ such that $0 \le \delta_i < \delta_i' < \bar{\ell}^i$ and $(\bar{\mathsf{Z}}^i_{\delta_i'}, \bar{\mathsf{Y}}^{i'}_{\delta_i'}) = (\mathsf{Z}^{i'}_{j'}, \mathsf{Y}^{i'}_{j'})$.
- $\mathtt{B13}$: $\exists (i,j) \in [q_e] \times [m^i]$ such that $|\{j \in [q_p] : \widehat{\mathsf{Z}}^j = \mathsf{Z}^i\}| \ge 2^{n-1}$.

Bad Transcript Analysis: For brevity we accumulate the above given conditions in certain compound events as follows:

- $\mathtt{Kcoll}$: $\mathtt{B1} \cup \mathtt{B2} \cup \mathtt{B3} \cup \mathtt{B4} \cup \mathtt{B5} \cup \mathtt{B6}$.
- $\mathtt{EEmatch}$: $\mathtt{B7} \cup \mathtt{B8}$.
- $\mathtt{EPmatch}$: $\mathtt{B9} \cup \mathtt{B10}$.
- $\mathtt{Chain}$: $\mathtt{B11} \cup \mathtt{B12}$.
- $\mathtt{EPKcoll}$: $\mathtt{B13}$.

Clearly, we have,

$$\Pr[\Theta_0 \in \Omega_{\mathsf{bad}}] = \Pr[\mathtt{Kcoll} \cup \mathtt{EEmatch} \cup \mathtt{EPmatch} \cup \mathtt{Chain} \cup \mathtt{EPKcoll}].$$

We bound the right hand side as follows:

1. *Bounding* $\Pr[\texttt{Kcoll}]$: First, $\Pr[\texttt{B1}]$, $\Pr[\texttt{B2}]$, and $\Pr[\texttt{B3}]$ can be easily bounded to at most $\sigma_e/2^\kappa$, $\sigma_d/2^\kappa$, and $q_p/2^\kappa$, respectively. This can be argued using the uniform randomness of K. Second, we bound $\Pr[\texttt{B4}|\neg\texttt{B3}]$ and $\Pr[\texttt{B5}|\neg\texttt{B3}]$ to at most $q_e/2^{n/2}$ and $q_d/2^{n/2}$, respectively. This can be argued using the fact that the least significant $n/2$-bit of Z_0^i must be $0^{n/2}$, whence we have at most $2^{n/2}$ options for Z_0^i and each option holds with at most 2^{-n} probability. Finally, we bound $\Pr[\texttt{B6}]$ to at most $\sigma_e\sigma_d/2^\kappa$, where we use the fact that $\mathsf{Z}_j^i = \bar{\mathsf{Z}}_{j'}^{i'}$ must lead to a κ-bit non-trivial equation (as $\mathsf{N}^i \neq \bar{\mathsf{N}}^{i'}$). Using union bound we have

$$\Pr[\texttt{Kcoll}] \leq \frac{\sigma_e + \sigma_d + q_p}{2^\kappa} + \frac{q_e + q_d}{2^{n/2}} + \frac{\sigma_e\sigma_d}{2^\kappa}.$$

2. *Bounding* $\Pr[\texttt{EEmatch}|\neg\texttt{Kcoll}]$: For any $i \neq i'$, we have a system of two equations $\mathsf{Z}_j^i = \mathsf{Z}_{j'}^{i'}$ and $\mathsf{I}_j^i = \mathsf{I}_{j'}^{i'}$, where $\mathsf{I} \in \{\mathsf{X}, \mathsf{Y}\}$. The first equation holds with at most $2^{-\kappa}$ probability and the second equation holds with at most 2^{-n} probability. We have 2 choices for I and at most σ_e^2 choices for (i,j) and (i',j'), which gives

$$\Pr[\texttt{EEmatch}] \leq \frac{2\sigma_e^2}{2^{\kappa+n}}.$$

3. *Bounding* $\Pr[\texttt{EPmatch}|\neg\texttt{Kcoll}]$: Here we can have two cases:
Case 1: First, suppose the primitive query occurs before the encryption query. Then, we have

$$\Pr[\texttt{EPmatch}|\neg\texttt{Kcoll}] \leq 2q_p\sigma_e/2^{n+\kappa}.$$

Case 2: Now, suppose the primitive query occurs after the encryption query. We handle condition B9. In this case we look at the number of multicollisions on X values across all encryption query blocks. Let

$$\mathsf{mcoll}(X) := |\{\mathsf{X}_j^i = X \ : \ (i,j) \in [q_e] \times [m^i]\}|, \text{ and } \mathsf{mcoll} = \max_X \mathsf{mcoll}(X).$$

Let $\texttt{Mcoll}$ denote the event that $\mathsf{mcoll} \geq n$. Then, we have

$$\Pr[\texttt{Mcoll}] \leq \frac{\sigma_e^n}{2^{n(n-1)}} = \left(\frac{\sigma_e}{2^n - 1}\right)^n \leq \frac{\sigma_e}{2^{n-1}}.$$

For a fixed primitive query there can be at most mcoll many (i,j) pairs. Thus, we have

$$\begin{aligned}\Pr[\texttt{EPmatch}|\neg\texttt{Kcoll}] &\leq \Pr[\texttt{Mcoll}] + \Pr[\texttt{EPmatch}|\neg(\texttt{Kcoll} \vee \texttt{Mcoll})] \\ &\leq \frac{\sigma_e}{2^{n-1}} + \frac{nq_p}{2^\kappa}.\end{aligned}$$

B10 condition can be handled similarly.
Since the two cases are exhaustive, we have

$$\Pr[\texttt{EPmatch}|\neg\texttt{Kcoll}] \leq \max\left\{\frac{2q_p\sigma_e}{2^{n+\kappa}}, \frac{4\sigma_e}{2^n} + \frac{2nq_p}{2^\kappa}\right\}.$$

4. *Bounding* $\Pr[\texttt{Chain}|\neg(\texttt{Kcoll} \vee \texttt{EEmatch} \vee \texttt{EPmatch})]$: Let $\texttt{FT} := \neg(\texttt{Kcoll} \vee \texttt{EEmatch} \vee \texttt{EPmatch})$.
Bound on $\Pr[\texttt{B11}|\texttt{FT}]$: B11 condition effectively means that there is a multi-chain $\mathcal{C}_{\bar{\mathsf{C}}^i_{\delta_i+1\ldots\bar{m}^i}, \bar{\mathsf{T}}^i}$ of length at most $\bar{\ell}^i - \delta_i \leq \bar{\ell}^i$. We make the following claim on the size of $\mathcal{C}_{\bar{\mathsf{C}}^i_{\delta_i+1\ldots\bar{m}^i}, \bar{\mathsf{T}}^i}$, i.e. $\Lambda_{\bar{\ell}^i}$.
Claim (1).

$$\Pr\left[\Lambda_{\bar{\ell}^i} \geq \bar{\ell}^i\frac{2\sqrt{n}q_p}{2^{n/2}} + \frac{2nq_p}{2^n} + \frac{2q_p}{2^n}\right] \leq 3/2^n.$$

Then, it is easy to see that

$$\Pr[\texttt{B11}|\texttt{FT}] \leq \frac{2\sqrt{n}q_p\sigma_d}{2^{\kappa+n/2}} + \frac{2nq_pq_d}{2^{\kappa+n}} + \frac{2q_pq_d}{2^{\kappa+n}}.$$

Bound on $\Pr[\texttt{B12}|\texttt{FT} \wedge \neg\texttt{B11}]$: For each decryption query index i, the condition B12 is possible for a unique encryption query index i', such that $\bar{\mathsf{N}}^i = \mathsf{N}^{i'}$. This is similar to $\texttt{B11}|\texttt{FT}$, although we have to consider multi-chains ending at each block of the said encryption query. Let

$$\boldsymbol{\mathcal{C}} := \bigcup_{\delta_i+2 \leq k \leq \ell_i - 1} \mathcal{C}_{\bar{\mathsf{C}}^i_{\delta_i+1\ldots k}},$$

and $\Lambda = |\boldsymbol{\mathcal{C}}|$. We make the following claim on Λ.

Claim (2).

$$\Pr\left[\Lambda \geq \bar{\ell}^i \frac{4\sqrt{n}q_p}{2^{n/2}} + \frac{4nq_p}{2^n} + \frac{2q_p}{2^n}\right] \leq 3/2^n.$$

Then, it is easy to see that

$$\Pr[\mathtt{B12}|\mathtt{FT} \wedge \neg\mathtt{B11}] \leq \frac{4\sqrt{n}q_p\sigma_d}{2^{\kappa+n/2}} + \frac{4nq_pq_d}{2^{\kappa+n}} + \frac{2q_pq_d}{2^{\kappa+n}}.$$

5. *Bounding* $\Pr[\mathtt{EPKcoll}|\neg(\mathtt{Kcoll} \vee \mathtt{EEmatch} \vee \mathtt{EPmatch} \vee \mathtt{Chain})]$: This event bounds the possibility of exhausting a particular encryption block key via primitive query. Let $\mathcal{K}$ denote the set of all primitive queries which are repeated at least 2^{n-1} times. Then we must have $|\mathcal{K}| \leq q_p/2^{n-1}$. Thus, some encryption block key falls in $\mathcal{K}$ with at most $q_p/2^{\kappa+n-1}$ probability, whence we have

$$\Pr[\mathtt{EPKcoll}|\neg(\mathtt{Kcoll} \vee \mathtt{EEmatch} \vee \mathtt{EPmatch} \vee \mathtt{Chain})] \leq \frac{2\sigma_e q_p}{2^{\kappa+n}}.$$

Lemma 1 summarizes the probability of realizing a bad transcript in the ideal world.

Lemma 1. *For $n \geq 16$, $\sigma_e, \sigma_d < 2^n$, and $q_p < 2^\kappa$, we have*

$$\Pr[\Theta_0 \in \Omega_{\mathsf{bad}}] \leq \frac{3\sigma_e\sigma_d + 3nq_p}{2^\kappa} + \frac{4\sigma_e}{2^n} + \frac{q_e + q_d}{2^{n/2}} + \frac{2\sigma_e^2 + 2\sigma_e q_p + 6nq_pq_d + 4q_pq_d}{2^{\kappa+n}} + \frac{6\sqrt{n}q_p\sigma_d}{2^{\kappa+n/2}} + \frac{6q_d}{2^n}.$$

Proof Sketch for Claim 1: The multi-chain $\mathcal{C}$ can be subdivided into three sets:

– $\mathcal{C}_{\mathsf{fwd}}$, the set of all chains constructed using forward queries only. $|\mathcal{C}_{\mathsf{fwd}}|$ can be bounded by the number of multicollisions on $\bar{\mathsf{T}}^i$. Particularly, we have

$$\Pr\left[|\mathcal{C}_{\mathsf{fwd}}| \geq \frac{nq_p}{2^n}\right] \leq q_p\left(\frac{e}{n}\right)^n \leq \frac{q_p}{2^{2n}} \leq \frac{1}{2^n}.$$

The above equation can be easily argued using proposition 1.

– $\mathcal{C}_{\mathsf{bck}}$, the set of all chains constructed using backward queries only. This can be bounded by the number of multicollisions on $\bar{\mathsf{Y}}^i_{\delta_i+1}$, in a similar fashion as $|\mathcal{C}_{\mathsf{bck}}|$. Particularly, we have

$$\Pr\left[|\mathcal{C}_{\mathsf{bck}}| \geq \frac{nq_p}{2^n}\right] \leq \frac{1}{2^n}.$$

– $\mathcal{C}_{\mathsf{fwd\text{-}bck}}$, the set of all chains, where each chain consists of both IC^+ and IC^- queries. Let N_k denote the maximum of the number of primitive queries with key K and the number of primitive queries with key $K \odot \alpha$. We say that a key K is good if $N_K \geq 2^{n-1}$. The number of bad keys is bounded by $q_p/2^{n-1}$. We bound $|\mathcal{C}_{\mathsf{fwd\text{-}bck}}|$ for *good* keys only. Specifically, we obtain the bound

$$\Pr\left[|\mathcal{C}_{\mathsf{fwd\text{-}bck}}| \geq \bar{\ell}^i \frac{2\sqrt{n}q_p}{2^{n/2}} + \frac{2q_p}{2^n}\right] \leq \frac{1}{2^n}.$$

Finally, we have

$$\Pr\left[\Lambda_{\bar{\ell}^i} \geq \bar{\ell}^i \frac{2\sqrt{n}q_p}{2^{n/2}} + \frac{2nq_p}{2^n} + \frac{2q_p}{2^n}\right] \leq \frac{3}{2^n}.$$

The proof of claim 2 follows a similar line of argument as the proof of claim 1, with some minor extensions.

4.2 Good Transcript Analysis

We fix a good transcript $(\omega_e, \omega_d, \omega_p)$. Consider the multiset $\mathcal{Z} := \{Z^i_j : (i,j) \in [q_e] \times [m^i]\}$. Let $(K_1, \ldots, K_s)$ denote the tuple of distinct keys in $\mathcal{Z}$ and λ_i be the multiplicity of K_i in $\mathcal{Z} \cup \{\widehat{Z}^j : j \in [q_p]\}$ for all $i \in [s]$.

IDEAL WORLD: In the ideal world we have

$$\Pr[\Theta_0 = (\omega_e, \omega_d, \omega_p)] = \Pr[\omega_p] \times \frac{1}{2^\kappa} \times \frac{1}{2^{n(\sigma_e+q_e)}},$$

where in the right hand side the second and third terms correspond to the sampling of key, and ciphertext and tag, conditioned on the primitive query-responses. Note that here the decryption transcript holds with probability 1.

REAL WORLD: In the real world we have

$$\Pr[\Theta_1 = (\omega_e, \omega_d, \omega_p)] = \Pr[\omega_p] \times \frac{1}{2^\kappa} \times \frac{1}{(2^n)_{q_e}} \times \frac{1}{\prod_{i=1}^{s} (2^n)_{\lambda_i}} \times \Pr[\omega_d | \omega_p, \omega_e],$$

Now, consider $\Pr[\omega_d|\omega_p, \omega_e]$. Let $\neg\omega_d$ denote the event that some decryption query-response is unrealizable, i.e. the real world returns a valid message. Let ω_d^i denote the event that the i-th decryption attempt succeeds. Then, we have

$$\begin{aligned}
\Pr[\omega_d|\omega_p, \omega_e] &= (1 - \Pr[\omega_d|\omega_p, \omega_e]) \\
&\geq \left(1 - \sum_{i \in [q_d]} \Pr[\omega_d^i|\omega_p, \omega_e]\right) \\
&\geq \left(1 - \frac{2\sigma_d(\sigma_e + q_p)}{2^{\kappa+n}} - \frac{2q_d}{2^n}\right).
\end{aligned}$$

Here we assume that $\sigma_e + q_p \leq 2^{n-1}$. Basically, the argument builds upon the fact that for each decryption query there exist a unique starting point which is a fresh input. Now, the first term corresponds to the probability that any intermediate point after the fresh input collides with some encryption block or primitive query. Given that, all the intermediate inputs are fresh, especially the tag generation input, the forgery succeeds with probability given in the second term. We finalize the probability of realizing a good transcript in lemma 2.

Lemma 2. *For any $\omega \in \Omega \setminus \Omega_{\mathsf{bad}}$, we have*

$$\frac{\Pr[\Theta_1 = \omega]}{\Pr[\Theta_0 = \omega]} \geq \left(1 - \frac{2\sigma_d(\sigma_e + q_p)}{2^{\kappa+n}} - \frac{2q_d}{2^n}\right).$$

The result follows by application of the main result of coefficients H technique, as given in Eq.(2), and lemmata 1 and 2. □

5 Conclusion

In this paper we proved the security of COMET-n for $n = \kappa$ under the ideal cipher model. In particular, we show that COMET-128 is secure while the data complexity is less than 2^{64} bytes and the offline computations, including block cipher evaluations, is less than 2^{119}.

Acknowledgments

This work is supported in part by the Ministry of Science and Technology, Israel, and the Department of Science and Technology, Government of India, DST/INT/ISR/P-20/2017.

Shay Gueron is supported by The Israel Science Foundation (grant No. 1018/16); NSF-BSF Grant 2018640; The BIU Center for Research in Applied Cryptography and Cyber Security, in conjunction with the Israel National Cyber Bureau in the Prime Minister's Office; The Center for Cyber Law & Policy at the University of Haifa in conjunction with the Israel National Cyber Directorate in the Prime Minister's Office.

The authors would like to thank Mustafa Khairallah for sharing his observations on bad conditions B4 and B5 (see section 4.1).

References

1. NIST: Lightweight cryptography (2018) Online: https://csrc.nist.gov/Projects/Lightweight-Cryptography. Accessed: August 01, 2019.

2. Gueron, S., Jha, A., Nandi, M.: COMET: Counter mode encryption with tag. Submission to NIST LwC Standardization Process (Round 1) (2019) Online: https://csrc.nist.gov/CSRC/media/Projects/Lightweight-Cryptography/documents/round-1/spec-doc/comet-spec.pdf. Access: August 01, 2019.
3. Chakraborty, B., MridulNandi: mixFeed. Submission to NIST LwC Standardization Process (Round 1) (2019) Online: https://csrc.nist.gov/CSRC/media/Projects/Lightweight-Cryptography/documents/round-1/spec-doc/mixFeed-spec.pdf. Access: August 01, 2019.
4. Iwata, T., Khairallah, M., Minematsu, K., Peyrin, T.: REMUS v1.0. Submission to NIST LwC Standardization Process (Round 1) (2019) Online: https://csrc.nist.gov/CSRC/media/Projects/Lightweight-Cryptography/documents/round-1/spec-doc/Remus-spec.pdf. Access: August 01, 2019.
5. Iwata, T., Khairallah, M., Minematsu, K., Peyrin, T., Sasaki, Y., Sim, S.M., Sun, L.: Thank Goodness It's Friday (TGIF) v1.0. Submission to NIST LwC Standardization Process (Round 1) (2019) Online: https://csrc.nist.gov/CSRC/media/Projects/Lightweight-Cryptography/documents/round-1/spec-doc/TGIF-spec.pdf. Access: August 01, 2019.
6. NIST: Announcing the ADVANCED ENCRYPTION STANDARD (AES). Fedral Information Processing Standards Publication FIPS 197, National Institute of Standards and Technology, U. S. Department of Commerce (2001)
7. Beaulieu, R., Shors, D., Smith, J., Treatman-Clark, S., Weeks, B., Wingers, L.: The SIMON and SPECK Lightweight Block Ciphers. In: Proceedings of the 52nd Annual Design Automation Conference, San Francisco, CA, USA, June 7-11, 2015. (2015) 175:1–175:6
8. Beaulieu, R., Shors, D., Smith, J., Treatman-Clark, S., Weeks, B., Wingers, L.: SIMON and SPECK: Block Ciphers for the Internet of Things. IACR Cryptology ePrint Archive **2015** (2015) 585
9. Koo, B., Roh, D., Kim, H., Jung, Y., Lee, D., Kwon, D.: CHAM: A family of lightweight block ciphers for resource-constrained devices. In: Information Security and Cryptology - ICISC 2017 - 20th International Conference, Seoul, South Korea, November 29 - December 1, 2017, Revised Selected Papers. (2017) 3–25
10. Rudin, W.: Real and Complex Analysis, 3rd Ed. McGraw-Hill, Inc. (1987)
11. Dworkin, M.: Recommendation for Block Cipher Modes of Operation – Methods and Techniques. NIST Special Publication 800-38A, National Institute of Standards and Technology, U. S. Department of Commerce (2001)
12. Chakraborti, A., Datta, N., Nandi, M., Yasuda, K.: Beetle Family of Lightweight and Secure Authenticated Encryption Ciphers. IACR Trans. Cryptogr. Hardw. Embed. Syst. **2018**(2) (2018) 218–241
13. Chakraborti, A., Datta, N., Nandi, M., Yasuda, K.: Beetle Family of Lightweight and Secure Authenticated Encryption Ciphers. IACR Cryptology ePrint Archive **2018** (2018) 805
14. Chakraborti, A., Iwata, T., Minematsu, K., Nandi, M.: Blockcipher-based authenticated encryption: How small can we go? In: Cryptographic Hardware and Embedded Systems - CHES 2017 - 19th International Conference, Taipei, Taiwan, September 25-28, 2017, Proceedings. (2017) 277–298
15. Chakraborti, A., Iwata, T., Minematsu, K., Nandi, M.: Blockcipher-based authenticated encryption: How small can we go? IACR Cryptology ePrint Archive **2017** (2017) 649
16. Patarin, J.: Etude de Générateurs de Permutations Basés sur les Schémas du DES. PhD thesis, Université de Paris (1991)
17. Patarin, J.: The "coefficients H" technique. In: Selected Areas in Cryptography - SAC '08. Revised Selected Papers. (2008) 328–345
18. Chen, S., Steinberger, J.P.: Tight security bounds for key-alternating ciphers. In: Advances in Cryptology - EUROCRYPT '14. Proceedings. (2014) 327–350
19. Mennink, B., Neves, S.: Encrypted davies-meyer and its dual: Towards optimal security using mirror theory. In: Advances in Cryptology - CRYPTO '17. Proceedings, Part III. (2017) 556–583