Two-Query Nonce-Respecting Forgeries on COLM_0

MiniMax

November 2026

Abstract

Authenticated encryption schemes designed for defense in depth aim to preserve confidentiality and integrity even under operational pitfalls such as nonce misuse. COLM was selected into the final portfolio of the CAESAR competition and formally specified in the *Journal of Cryptology* (2024) with two recommended modes: the primary single-tag mode COLM_0 and the intermediate-tag mode COLM_{127}, claiming 64-bit integrity for both. Prior cryptanalysis of COLM_0 either required 2^{64} birthday-bound encryption queries or relied on unverified decryption (INT-RUP) queries outside COLM_0's security claims (Datta et al., *ToSC* 2017), leaving its integrity in the standard nonce-respecting, encryption-only (INT-CTXT) model seemingly intact.

In this paper, we show that COLM_0 fails to achieve its claimed integrity even against a standard nonce-respecting, encryption-only adversary. Intuitively, because COLM_0's input masking is independent of the nonce and its decryption checksum cancels intermediate plaintext changes, encrypting two messages with a shared prefix under distinct nonces induces a structured linear difference in the chaining state that can be canceled during decryption by splicing a small subset of ciphertext blocks. Building on this observation, we construct two deterministic two-query forgery families on COLM_0 using zero decryption queries. We further show why COLM_{127} resists two-query splicing, evaluate lightweight mode-level mitigations, pinpoint the corresponding gaps in both published security proofs, and experimentally verify all results across the official AES-128 and AES-256 implementations.

1 Introduction

Authenticated Encryption with Associated Data (AEAD) schemes simultaneously guarantee the confidentiality and authenticity of plaintext messages alongside authenticated header data (AD). Classical single-pass AEAD modes such as OCB [11] and AES-GCM achieve optimal efficiency when every encryption query uses a fresh nonce, yet their security degrades catastrophically if a nonce is accidentally repeated.

To provide *defense in depth* against such implementation pitfalls without sacrificing single-pass online processing or parallelizability, Bogdanov, Datta, Luykx, Mennink, Nandi, Tischhauser, and collaborators developed the *Encrypt–Linear-Mix–Encrypt* (ELE) design paradigm. In an ELE mode based on an n-bit blockcipher $E_K : \{0,1\}^n \to \{0,1\}^n$, each n-bit plaintext block $M[i]$ undergoes two blockcipher invocations separated by a stateful linear mixing layer $\rho : \mathbb{F}_{2^n} \times \mathbb{F}_{2^n} \to \mathbb{F}_{2^n} \times \mathbb{F}_{2^n}$, coupled via $\mathbb{F}_{2^n}$-linear XEX-style input and output masks $\Delta_M[i]$ and $\Delta_C[i]$.

The lineage of COLM evolved across four closely related constructions:

1. COPA **(ASIACRYPT 2013 [1]):** Introduced parallelizable single-pass nonce-misuse resistance up to the longest common plaintext prefix (online privacy) using simple XOR mixing $(X, W) \mapsto (X \oplus W, X \oplus W)$. Tag generation uses a plaintext XOR checksum $M^*[1] \oplus \cdots \oplus M^*[\ell]$, while incomplete final blocks were handled via XLS (whose security claims were subsequently revisited in [14]).

2. ELmE **(ACISP 2014 [8]):** Replaced XOR mixing with the invertible finite-field linear mixing map $\rho_{1,3,1,2}(X, W) = (X \oplus 3W, X \oplus 2W)$ over $\mathbb{F}_{2^n}$. It also introduced optional intermediate tags to defend against Release of Unverified Plaintext (RUP) adversaries.

3. ELmD **(CAESAR Round 1, 2014 [9]):** Retained ELmE's invertible linear mixing layer $\rho_{1,3,1,2}$ while replacing the lower-layer blockcipher encryption call with decryption E_K^{-1}; during decryption, each ciphertext block passes through E_K, the inverse mixing map $(Y, W) \mapsto (Y \oplus 3W, Y \oplus W)$, and finally E_K^{-1} on its data output, so that encryption and decryption share a first-E_K / second-E_K^{-1} blockcipher pipeline.

4. COLM **(CAESAR Round 2/3, 2015–2019 [3, 4]; *J. Cryptology* 2024 [5]):** Combined the COPA and ELmD designs during Round 2 of the CAESAR competition (*COmbining* COPA *and* ELmD). In February 2019, COLM was selected into the CAESAR Final Portfolio as the second choice for Use Case 3 (*Defense in Depth*).

In the COLM v1 specification [3, Section 1.2, p. 2] and the 2024 *Journal of Cryptology* specification [5, Section 3.1, pp. 9–14; Section 5.1, p. 27], the designers specify two recommended modes of operation on block size $n = 128$ and tag length $\ell_\tau = 128$ bits, instantiated with AES-128 (128-bit key) in COLM v1 and with both AES-128 and AES-256 (256-bit key) in [5]:

- **Primary Mode** COLM_0 **($(\tau, \ell_\tau) = (0, 128)$):** Emits *no intermediate tags* ($\tau = 0$) for general-purpose applications, incurring a standard 16-byte expansion ($|C| = |M| + 16$ bytes), with AES instantiations AES-COLM_0-128 and AES-COLM_0-256 [5].

- **Intermediate-Tag Mode** COLM_{127} **($(\tau, \ell_\tau) = (127, 128)$):** Interleaves a 128-bit intermediate tag $T[j] = E_K(W[127j]) \oplus \Delta_3[j_{\text{tag}}]$ after every $\tau = n - 1 = 127$ message blocks, instantiated as AES-COLM_{127}-128 and AES-COLM_{127}-256 [5].

For *both* COLM_0 and COLM_{127}, the security claims in [3, Section 3, p. 10] and Theorem 2 in [5, Section 4, p. 15] assert $n/2 = 64$-bit security (2^{64} query/block complexity)—covering all confidentiality and integrity goals under both unique and repeated nonces.

Existing Cryptanalysis. Prior to our work, the published cryptanalytic literature on COLM and its predecessor constructions established the following results (summarized in Table 1):

- **Birthday-Bound Mode-Level Attacks:** Building on birthday-bound mask-recovery ($L = E_K(0)$), state-collision, and reduced-round ELmD attacks on COPA [12] and ELmD [2], Vaudenay and Vizár [16, Section 18] and Forler et al. [10] analyzed existential and multiple forgeries on COLM in the encryption-only and nonce-misuse settings, while Ulusoy et al. [15] and Chakraborty and Nandi [6] studied universal forgery and chosen-ciphertext (CCA) plaintext-recovery attacks on COLM and its generalized linear-mixing variants. All of these mode-level attacks require $\Omega(2^{n/2}) = 2^{64}$ queries, matching the $\Omega(2^{n/2})$ birthday barrier for tweakable Even-Mansour/XEX masking [13]. Vaudenay and Vizár explicitly note that *"none of the attacks we present violates the security claims of any of the CAESAR candidates"* [16].

- **Constant-Query Forgery on ELmE under Nonce Reuse (Datta and Nandi [8, Section 5.2]):** On the predecessor scheme ELmE with intermediate tags at stride $k \geq n$, Datta and Nandi observed that two encryption queries sharing the same initial value (i.e., repeating both nonce and associated data) and differing in block 1 can be spliced via a degree-n primitive polynomial identity (illustrated at $n = 4$ with $x^4 + x^3 + 1$) to align the chaining state with the second query before an intermediate tag. Unlike our attacks, this message-difference mechanism requires repeating the nonce ($\Delta\mathsf{IV} = 0$): under distinct nonces, varying the first message block alongside $\Delta\mathsf{IV} \neq 0$ introduces two independent differences that cannot be simultaneously canceled (Proposition 1).

Table 1: Comparison of forgery attacks against COLM ($n = 128$, claimed integrity: 2^{64} queries/blocks).

Reference	Target	Nonce	Oracle Model	Queries	Blocks
Vaudenay and Vizár [16]	$\mathsf{COLM}_0, \mathsf{COLM}_{127}$	Unique	Encryption only	2^{64}	2^{65}
Datta et al. [7][a]	COLM_0	Repeated	Enc. + RUP Dec.	2^{10}	$2^{18.6}$
Datta et al. [7][b]	COLM_0	Unique	Enc. + RUP Dec.	2^{8}	2^{22}
This Work (Difference Fade)[c]	COLM_0	**Unique**	**Encryption only**	**2**	**262**
This Work (State Transfer)[d]	COLM_0	**Unique**	**Encryption only**	**2**	**260**

[a]Section 5 of [7]; requires repeated nonces and unverified decryption (INT-RUP, outside COLM_0's security claim).
[b]Section 6 of [7]; requires 1 encryption query and 256 unverified decryption (INT-RUP) queries of 2^{14} blocks each.
[c]Theorem 1 ($\ell_{\min} = 130$ plaintext blocks plus 1 tag block per query; $2(\ell_{\min} + 1) = 262$ blocks total).
[d]Theorem 2 ($\ell_{\min} = 129$ plaintext blocks plus 1 tag block per query; $2(\ell_{\min} + 1) = 260$ blocks total).

- **Unverified-Decryption (INT-RUP) Attacks on COLM_0 (Datta, Luykx, Mennink, and Nandi [7]):** In *ToSC* 2017, four of COLM's designers analyzed the integrity of COLM_0 under Release of Unverified Plaintext (INT-RUP). To respect the unique-nonce restriction on forward queries, the authors confined their state-collision procedure to decryption queries [7, Section 6, p. 152]: *"to comply with the condition that nonces should never be repeated in encryption queries, queries are only made in inverse direction."* Their resulting nonce-respecting forgery requires one encryption query and $2n = 256$ ($2^{8.0}$) unverified decryption (RUP) queries (each of $(n+1)n = 2^{14.0}$ blocks, for $2^{22.0}$ blocks in total). Because COLM makes no RUP security claim for COLM_0, this attack falls outside COLM_0's stated security model.

Our Contributions. As summarized in Table 1, we establish that COLM_0 admits a deterministic, constant-query existential forgery in the standard *nonce-respecting, encryption-only* (INT-CTXT) security model:

1. **Two-Query Nonce-Respecting Encryption-Only Forgeries on COLM_0 (Section 3):** We show that the inverse-query assumption in [7, Section 6] is unnecessary. By querying the encryption oracle on two messages sharing a 129-block (2,064-byte) prefix under two *distinct* nonces $N_1 \neq N_2$ (or distinct associated data $A_1 \neq A_2$), the upper-layer blockcipher outputs cancel ($X^{(1)}[i] = X^{(2)}[i]$ for $1 \leq i \leq 129$) because $\Delta_M[i] = 2^i L$ is independent of (N, A). This leaves a single unknown differential $\Delta\mathsf{IV} = \mathsf{IV}_1 \oplus \mathsf{IV}_2$, giving rise to two 2-query encryption-only forgery families with tight minimum block lengths:
   - *Difference Fade (Theorem 1):* Splicing the 5 blocks $S_{\text{fade}} = \{1, 2, 3, 8, 129\} = \text{supp}(f)+1$ of the second ciphertext into the first at message length $\ell \geq 130$ blocks (2,080 bytes, where $\ell_{\min} = 130$ is tight) forces the decryption chaining-state difference to vanish identically at block 129 via $1 \oplus x \oplus x^2 \oplus x^7 \oplus x^{128} = f(x) = 0$ in $\mathbb{F}_{2^{128}}$. The first query's unmodified tag block $C^{(1)}[\ell + 1]$ then verifies with probability 1.
   - *State Transfer and Tag Transplant (Theorem 2):* Splicing the six-block prefix set $\text{supp}(3^{-1}x^{128}) + 1 = \{1, 3, 4, 5, 6, 7\}$ together with a trailing block run $\{m, \ldots, \ell + 1\}$ for $m \geq 129$ ($\ell_{\min} = 129$ blocks) aligns the decryption chaining state with that of the second query ($W_{\text{mix}}[m-1] = W^{(2)}[m-1]$), allowing the second query's tag block $C^{(2)}[\ell + 1]$ to be transplanted directly.
2. **Diagnosis of Both Official Security Proofs (Section 4):** At the parameters of our two-query adversary, the H-coefficient proofs in the 2024 *Journal of Cryptology* specification [5] and the 2016 CAESAR Round-3 addendum [4] claim forgery bounds of $2^{-105.92}$ and $2^{-107.56}$,

whereas our attack succeeds with probability 1. We trace this gap to two oversights in the proofs: (i) the good-transcript forgery bound assumes that a fresh intermediate decryption block randomizes the final checksum block, overlooking that decryption checksum telescoping makes the tag check independent of all intermediate plaintext blocks; and (ii) the bad-event analysis treats the mask powers $2^r \in \mathbb{F}_{2^{128}}$ as linearly independent for arbitrary message lengths, overlooking that the degree-128 field polynomial $f(x) = 0$ creates non-trivial linear dependencies once $\ell \geq 129$.

3. **Degree Obstruction for COLM_{127} and Mitigation Analysis (Section 5):** Extending the $\tau = n-1 = 127$ degree-bound rationale of [8, Lemma 9] and [7, Remark 1] to cross-nonce $\Delta\mathsf{IV}$ splicing, we prove that $\deg(3^{-1}x^\tau \bmod f(x)) = 127 > \tau - 1$ for all $1 \leq \tau \leq 127$. This shows that COLM_{127} ($\tau = 127$) prevents both two-query splicing families with a safety margin of a single block ($\tau^* = 128$). We also evaluate lightweight mode-level mitigations and their overheads.

2 Specification and Key Processes of COLM

We follow the notation of the COLM specifications [3, 5] and the security proof addendum [4]. Before introducing the underlying field algebra, we summarize the operational interface and two-stage architecture of COLM_0.

2.1 Operational Setting and Two-Layer ELE Architecture

Under a secret blockcipher key K (128 bits for AES-128 or 256 bits for AES-256), COLM_0 provides nonce-based authenticated encryption with associated data (AEAD) built from a 128-bit blockcipher $E_K : \{0,1\}^{128} \to \{0,1\}^{128}$ (with inverse E_K^{-1}). Encryption $\mathsf{COLM}_0\text{-}\mathsf{Enc}_K(N, A, M^*)$ accepts a 64-bit public message number (nonce) $N \in \{0,1\}^{64}$, variable-length associated data $A \in \{0,1\}^*$ partitioned into a blocks $A[1], \ldots, A[a]$ (with 10^* padding when $A[a]$ is incomplete), and a plaintext M^*. Throughout our analysis, we consider block-aligned plaintexts of $\ell \geq 1$ complete 128-bit blocks $M^*[1], \ldots, M^*[\ell]$ (incomplete final blocks follow identically under 10^* padding and 7^2 mask scaling). Encryption outputs $\ell + 1$ blocks $C[1 \ldots \ell + 1]$, consisting of an ℓ-block ciphertext $C[1 \ldots \ell]$ and a 128-bit authentication tag $T = C[\ell + 1]$. Conversely, verified decryption $\mathsf{COLM}_0\text{-}\mathsf{Dec}_K(N, A, C[1 \ldots \ell + 1])$ either returns the authenticated plaintext $M^*[1 \ldots \ell]$ if tag verification succeeds, or outputs the rejection symbol $\perp$ without releasing any unverified plaintext.

As illustrated in Figure 1(a) (adapted from [3, Figure 1.1, p. 4] and [5, Figure 1, p. 10]), COLM_0 processes every query in two sequential stages:

1. **Stage 1 — Parallel Nonce and Associated-Data Initialization (IV Generation):** In the left dashed box of Figure 1(a), the 64-bit nonce N is concatenated with the 64-bit parameter descriptor param (0^{64} in COLM_0) to form an initial header block $A[0] = N \parallel \mathsf{param}$. A PMAC-style parallel layer masks $A[0]$ and each associated-data block $A[1], \ldots, A[a]$ with distinct subkey masks $\Delta_A[0 \ldots a]$, encrypts each masked block independently under E_K, and XORs all $a + 1$ outputs to form a 128-bit initial value IV. This IV initializes the horizontal chaining register entering the message pipeline: $W[0] \leftarrow \mathsf{IV}$.

2. **Stage 2 — Encrypt–Linear-Mix–Encrypt (ELE) Message and Tag Pipeline:** Prior to encryption, COLM_0 maps the ℓ plaintext blocks $M^*[1 \ldots \ell]$ into $\ell + 1$ internal 128-bit input blocks. It retains $M[i] = M^*[i]$ for $1 \leq i \leq \ell - 1$, replaces the ℓ-th block with the full plaintext XOR checksum $M[\ell] = \bigoplus_{k=1}^{\ell} M^*[k]$, and appends a duplicate checksum block $M[\ell + 1] = M[\ell]$ dedicated to tag generation. In the right dashed box of Figure 1(a), each column $i \in \{1, \ldots, \ell + 1\}$ passes top-to-bottom through two keyed

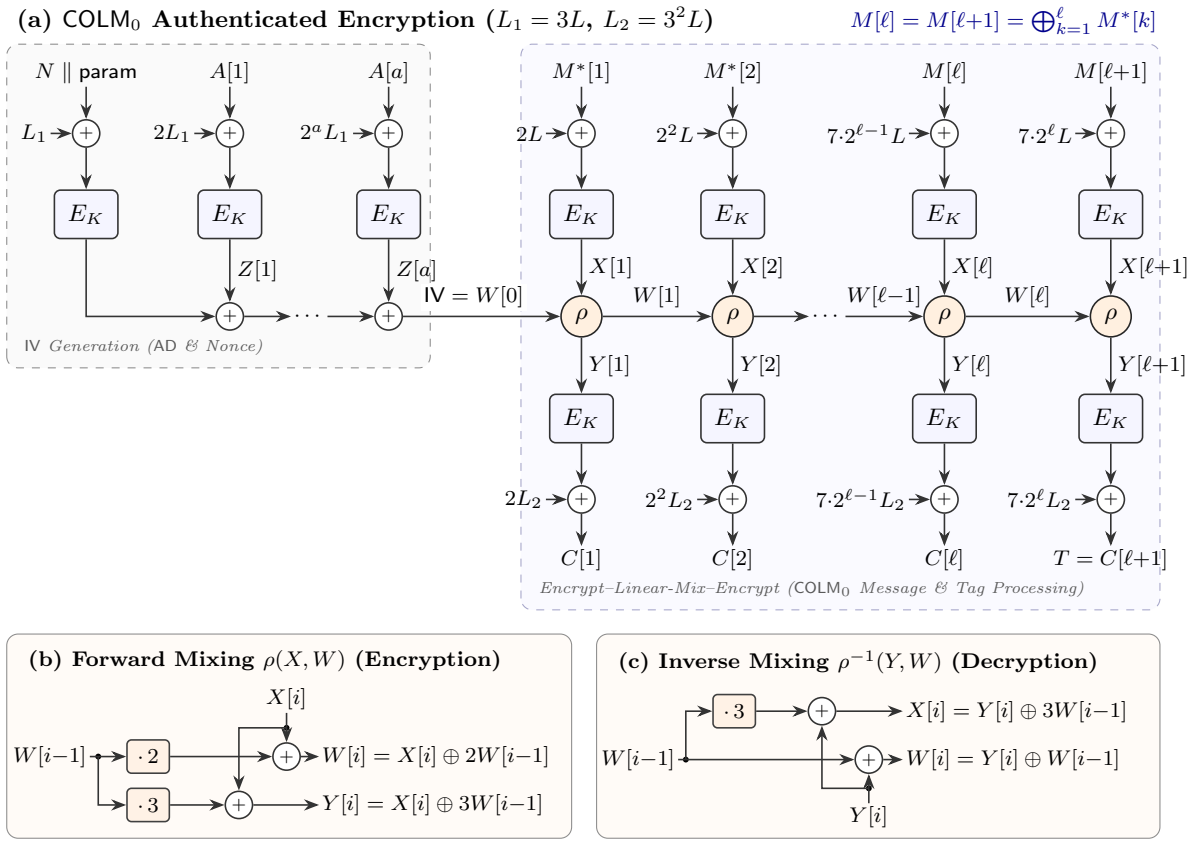

Figure 1: Overall Encrypt–Linear-Mix–Encrypt architecture of COLM$_0$ (adapted from [3, Figure 1.1] and [5, Figure 1]). **(a)** Parallel AD/nonce initialization produces the initial chaining state $\mathsf{IV} = W[0]$; each message/tag column i then traverses an upper blockcipher layer (E_K), a stateful linear mixing layer (ρ) threaded horizontally by $W[0] \to W[1] \to \cdots \to W[\ell]$, and a lower blockcipher layer (E_K). **(b)** Forward linear mixing ρ in encryption. **(c)** Inverse linear mixing ρ^{-1} in decryption, where the chaining update simplifies to $W[i] = W[i-1] \oplus Y[i]$.

blockcipher layers separated by a stateful linear mixing layer. Meanwhile, a 128-bit *chaining state* $W[0] \to W[1] \to \cdots \to W[\ell]$ propagates horizontally from left to right:

- *Upper blockcipher layer:* This layer comprises the top row of E_K invocations on the message path. At column i, the internal block $M[i]$ is masked with an XEX plaintext mask $\Delta_M[i]$ and encrypted via E_K to produce the *upper-layer output* $X[i] = E_K(M[i] \oplus \Delta_M[i])$. Notice that this layer operates strictly block-by-block in parallel and receives no input from (N, A) or earlier blocks.
- *Middle linear mixing layer (ρ) and chaining state (W):* Positioned between the two blockcipher rows, the invertible linear map ρ couples column i's upper-layer output $X[i]$ with the *incoming chaining state* $W[i-1]$. That state is the 128-bit horizontal state register emitted by the preceding column $i-1$, initialized at $i = 1$ to $W[0] = \mathsf{IV}$. Evaluating $(Y[i], W[i]) = \rho(X[i], W[i-1])$ simultaneously computes the mixed vertical block $Y[i]$ fed into column i's second blockcipher call and the *updated chaining state* $W[i]$ passed horizontally as the incoming state to column $i+1$. This horizontal state wire is the sole mechanism diffusing the (N, A)-dependent IV and prior message blocks $M[1 \ldots i-1]$ into subsequent ciphertext blocks and the tag.
- *Lower blockcipher layer:* This layer comprises the bottom row of E_K invocations on the message path. At column i, the mixed state $Y[i]$ is encrypted through a second E_K call and XORed with a ciphertext mask $\Delta_C[i]$ to output $C[i] = E_K(Y[i]) \oplus \Delta_C[i]$. This

produces ciphertext blocks for $1 \le i \le \ell$ and the authentication tag $T = C[\ell+1]$ at $i = \ell + 1$.

Inverse Dataflow During Decryption. When decrypting $C[1 \dots \ell+1]$ under (N, A), the receiver first recomputes the identical initial chaining state $W[0] = \mathsf{IV}$ via Stage 1. It then traverses columns $i = 1, \dots, \ell$ from bottom to top. At each column it inverts the *lower blockcipher layer* via $Y[i] = E_K^{-1}(C[i] \oplus \Delta_C[i])$. It next combines $Y[i]$ with the *incoming chaining state* $W[i-1]$ through the *inverse linear mixing map* $(X[i], W[i]) = \rho^{-1}(Y[i], W[i-1])$ (Figure 1(c)) to recover the upper-layer state $X[i]$ while advancing the chaining state to $W[i]$. Finally it inverts the *upper blockcipher layer* via $\tilde{M}[i] = E_K^{-1}(X[i]) \oplus \Delta_M[i]$. At column $\ell + 1$, the receiver evaluates a forward tag-verification pass from the recovered checksum and $W[\ell]$ to test whether $\tilde{C}[\ell+1] = C[\ell+1]$.

2.2 Field Arithmetic, Masking Schedule, and Linear Mixing

With the architectural dataflow established, we state the algebraic definitions of the field arithmetic, the XEX masks, and the mixing maps ρ and ρ^{-1}. Every 128-bit block ($n = 128$) is identified with an element of the binary extension field

$$\mathbb{F}_{2^{128}} = \mathbb{F}_2[x]/\big(f(x)\big), \qquad f(x) = x^{128} + x^7 + x^2 + x + 1. \tag{1}$$

In the integer notation of [3, Section 1.1, pp. 1–2], the field element $x \in \mathbb{F}_{2^{128}}$ is written as 2, $x + 1$ is written as 3, and $x^2 + x + 1 = 3^2 \oplus 2$ is written as 7. Throughout this paper, addition in $\mathbb{F}_{2^{128}}$ is bitwise XOR ($\oplus$), and juxtaposition or $(\cdot)$ denotes multiplication in $\mathbb{F}_{2^{128}}$ modulo $f(x)$.

Subkey and Masking Schedule $(\Delta_A, \Delta_M, \Delta_C)$. From the secret key K, COLM derives a single secret mask base $L \leftarrow E_K(0^{128}) \in \mathbb{F}_{2^{128}}$. It then initializes three domain-separated base masks [3, Section 1.4, pp. 4–5], [5, Figures 2–3, pp. 11–12], one each for Stage 1 header hashing, the upper blockcipher layer, and the lower blockcipher layer:

$$\Delta_0 \leftarrow 3 \cdot 2^{-1} L, \qquad \Delta_1 \leftarrow L, \qquad \Delta_2 \leftarrow 3^2 L. \tag{2}$$

On the header block $A[0] = N \parallel \mathsf{param}$, Stage 1 uses $\Delta_A[0] = 2\Delta_0 = 3L$; for non-empty associated data ($a = \lceil |A|/128 \rceil \ge 1$), the AD mask multiplies by 2 (i.e., x) on every non-final block ($1 \le j < a$) and on a complete final block ($|A| = 128a$), and by 7 on an incomplete (10^*-padded) final block ($|A| < 128a$). In Stage 2, the message and ciphertext masks (Δ_M, Δ_C) multiply by 2 on every non-final message block ($1 \le i \le \ell - 1$), by 7 on a complete final message block ($i = \ell$), by 7^2 on an incomplete (10^*-padded) final message block ($i = \ell$), and by 2 again on the tag block ($i = \ell + 1$). Thus, for a message of ℓ complete 128-bit blocks, the upper-layer plaintext masks $\Delta_M[i]$ and lower-layer ciphertext masks $\Delta_C[i]$ are:

$$\Delta_M[i] = \begin{cases} 2^i L & \text{for } 1 \le i \le \ell - 1, \\ 7 \cdot 2^{\ell-1} L & \text{for } i = \ell, \\ 7 \cdot 2^{\ell} L & \text{for } i = \ell + 1 \text{ (tag block)}, \end{cases} \qquad \Delta_C[i] = 3^2\, \Delta_M[i] \quad (1 \le i \le \ell + 1). \tag{3}$$

By (3), $\Delta_M[i]$ *and* $\Delta_C[i]$ *depend solely on the secret key* K *(via* L*), the block index* i*, and the message length* ℓ*. They depend neither on the nonce* N *nor on the associated data* A*.*

The Linear Mixing Map ρ and Its Inverse ρ^{-1}. At column i of Stage 2, let $X[i] \in \mathbb{F}_{2^{128}}$ be the output of the upper blockcipher layer and $W[i-1] \in \mathbb{F}_{2^{128}}$ the incoming horizontal chaining state from column $i - 1$ (with $W[0] = \mathsf{IV}$). Let $Y[i] \in \mathbb{F}_{2^{128}}$ be the input to the lower blockcipher layer and $W[i] \in \mathbb{F}_{2^{128}}$ the updated horizontal chaining state passed to column $i + 1$.

As depicted in Figures 1(b) and 1(c), the forward mixing map $\rho(X, W) = (Y, W')$ and the inverse mixing map $\rho^{-1}(Y, W) = (X, W')$ [3, Section 1.3, p. 3] used during encryption and decryption, respectively, are defined over $\mathbb{F}_{2^{128}}$ by:

$$\rho(X, W) = \begin{pmatrix} Y \\ W' \end{pmatrix} = \begin{pmatrix} 1 & 3 \\ 1 & 2 \end{pmatrix} \begin{pmatrix} X \\ W \end{pmatrix} = \begin{pmatrix} X \oplus 3W \\ X \oplus 2W \end{pmatrix}, \tag{4}$$

where $3 \oplus 2 = (x + 1) \oplus x = 1$ holds in $\mathbb{F}_{2^{128}}$. Substituting $X = Y \oplus 3W$ into $W' = X \oplus 2W$ therefore cancels the multiplier on W, giving

$$\rho^{-1}(Y, W) = \begin{pmatrix} X \\ W' \end{pmatrix} = \begin{pmatrix} 1 & 3 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} Y \\ W \end{pmatrix} = \begin{pmatrix} Y \oplus 3W \\ Y \oplus W \end{pmatrix}. \tag{5}$$

Whereas encryption advances the chaining state via field multiplication by 2 ($W[i] = X[i] \oplus 2W[i-1]$), decryption (5) reduces the horizontal chaining update to the plain running XOR of the lower-layer intermediate states:

$$W[i] = W[i-1] \oplus Y[i] = \mathsf{IV} \oplus \bigoplus_{j=1}^{i} Y[j] \qquad \forall i \geq 1. \tag{6}$$

2.3 Complete Algorithmic Flow of COLM_0

To complement the encryption diagram in Figure 1(a), Figure 2 illustrates the verified decryption and tag-checking architecture $\mathsf{COLM}_0\text{-}\mathsf{Dec}_K(N, A, C[1 \dots \ell + 1])$. Algorithm 1 gives the joint pseudocode of both procedures [3, Sections 1.4–1.5, pp. 4–8]. In Figure 2, Stage 1 recomputes $\mathsf{IV} = W[0]$ in the forward direction (E_K). Message columns $i = 1, \dots, \ell$ run bottom-to-top through E_K^{-1} and ρ^{-1} while advancing $W[i] = W[i-1] \oplus Y[i]$ horizontally. Following [3, Section 1.5.3, p. 8] and [5, Figure 3, p. 12], column $\ell + 1$ then runs top-to-bottom from CS and $W[\ell]$ via E_K and ρ to recompute $\tilde{C}[\ell+1]$ and test $\tilde{C}[\ell+1] \stackrel{?}{=} C[\ell+1]$ before releasing $\tilde{M}^*[1 \dots \ell]$ or returning $\perp$.

2.4 Checksum Telescoping in COLM_0 Decryption

Inspecting Step 2 ($i = \ell$) and Step 3 of $\mathsf{COLM}_0\text{-}\mathsf{Dec}$ in Algorithm 1 (see [3, Sections 1.5.1 and 1.5.3, pp. 7–8] and [5, Figure 3, p. 12]) reveals a key structural property of COLM_0:

Lemma 1 (Checksum Telescoping in COLM_0 Decryption)**.** *For any ciphertext $C[1 \dots \ell+1]$ with a complete final block ($|C[\ell+1]| = 128$ bits), the checksum value CS fed into the tag-verification block at Step 3 of $\mathsf{COLM}_0\text{-}\mathsf{Dec}$ satisfies*

$$\mathsf{CS} = \bigoplus_{i=1}^{\ell} \tilde{M}^*[i] = \left(\bigoplus_{i=1}^{\ell-1} \tilde{M}^*[i] \right) \oplus \left(\tilde{M}[\ell] \oplus \bigoplus_{i=1}^{\ell-1} \tilde{M}^*[i] \right) \equiv \tilde{M}[\ell], \tag{7}$$

where $\tilde{M}[\ell] = E_K^{-1}\big(E_K^{-1}(C[\ell] \oplus \Delta_C[\ell]) \oplus 3W[\ell-1]\big) \oplus \Delta_M[\ell]$. Consequently, the recomputed tag block $\tilde{C}[\ell+1]$ is a deterministic function of $(K, \ell, C[\ell], W[\ell-1])$ alone and does not *depend on the decrypted plaintext blocks $\tilde{M}^*[1], \dots, \tilde{M}^*[\ell-1]$.*

Proof. During Encryption (Step 2), the ℓ-th internal block is defined as $M[\ell] = \bigoplus_{i=1}^{\ell} M^*[i]$ and the $(\ell+1)$-th internal block is set to $M[\ell+1] = M[\ell]$. During Decryption (Step 2), the receiver recovers the first $\ell - 1$ plaintext blocks $\tilde{M}^*[i] = \tilde{M}[i]$ $(1 \leq i \leq \ell - 1)$ and defines the ℓ-th plaintext block as $\tilde{M}^*[\ell] := \tilde{M}[\ell] \oplus \bigoplus_{i=1}^{\ell-1} \tilde{M}^*[i]$. Therefore, the XOR sum $\bigoplus_{i=1}^{\ell} \tilde{M}^*[i]$ used as $M[\ell+1]$ in Step 3 cancels the partial checksum $\bigoplus_{i=1}^{\ell-1} \tilde{M}^*[i]$ identically in characteristic 2, leaving $\mathsf{CS} \equiv \tilde{M}[\ell]$. Indeed, both official specifications ([3, Section 1.5.1, p. 7] and [5, Figure 3,

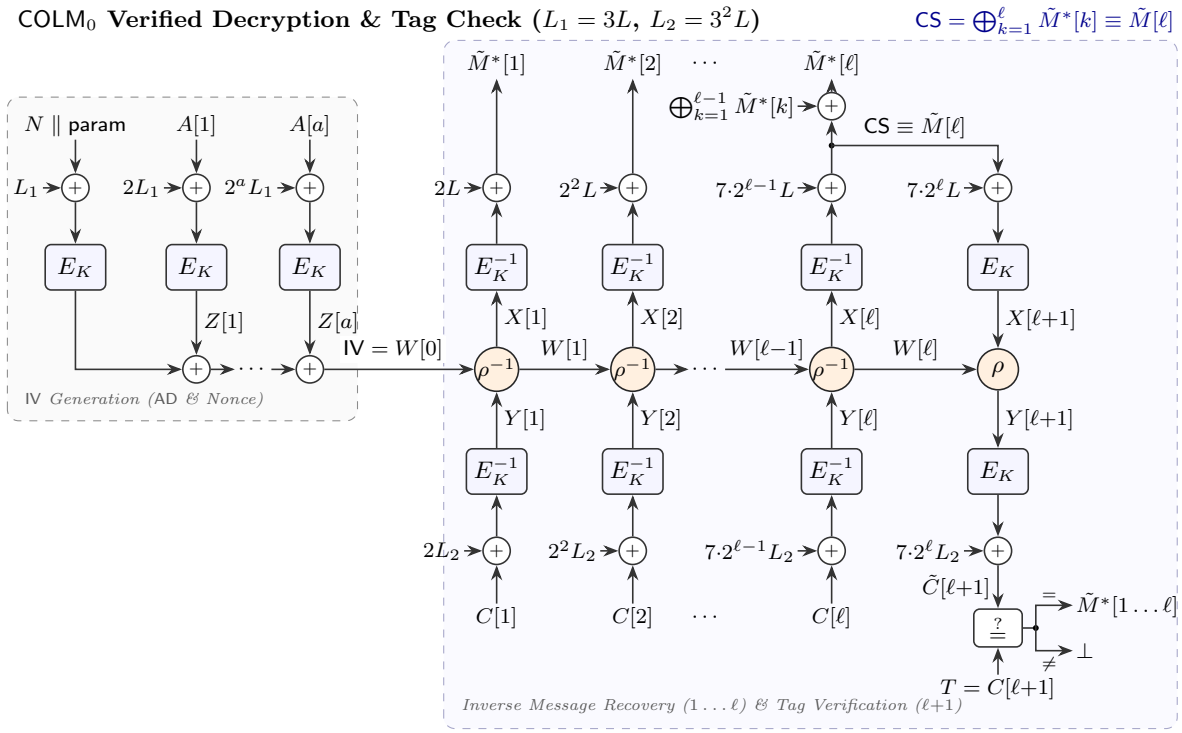

Figure 2: Complete verified decryption and tag-checking architecture of COLM_0 (COLM_0-Dec, mirroring Figure 1(a)). Stage 1 recomputes $\mathsf{IV} = W[0]$ via forward E_K calls; columns $1, \ldots, \ell$ invert the lower and upper blockcipher layers (E_K^{-1}) and linear mixing (ρ^{-1}) bottom-to-top while advancing $W[i] = W[i-1] \oplus Y[i]$. At column ℓ, $\tilde{M}[\ell]$ both unmasks the final plaintext block $\tilde{M}^*[\ell]$ and feeds $\mathsf{CS} = \bigoplus_{k=1}^{\ell} \tilde{M}^*[k] \equiv \tilde{M}[\ell]$ (Lemma 1) into the forward tag-verification column $\ell + 1$, which releases $\tilde{M}^*[1 \ldots \ell]$ if and only if $\tilde{C}[\ell + 1] = C[\ell + 1]$.

p. 12]) as well as the parallel AES-NI implementations (`aesni4`, `aesni8`) state this identity directly by assigning $M[\ell + 1] \leftarrow \tilde{M}[\ell]$ during decryption, whereas the portable reference C implementation (`colm0v1/ref`) accumulates $\mathsf{CS} = \bigoplus_{i=1}^{\ell} \tilde{M}^*[i]$ which algebraically telescopes to the identical value $\tilde{M}[\ell]$ by (7). Since $W[\ell] = W[\ell - 1] \oplus Y[\ell]$ and both $Y[\ell]$ and $\tilde{M}[\ell]$ are determined by $(K, \ell, C[\ell], W[\ell - 1])$, the tag verification test $\tilde{C}[\ell + 1] = C[\ell + 1]$ depends on blocks $C[1 \ldots \ell - 1]$ *solely through the chaining state* $W[\ell - 1]$. □

3 Two-Query Nonce-Respecting Forgeries on COLM_0

Consider modifying a ciphertext $C^{(1)}[1 \ldots \ell + 1]$ on a subset of block indices $S \subseteq \{1, \ldots, \ell - 1\}$ while leaving $(C^{(1)}[\ell], C^{(1)}[\ell + 1])$ intact. By Lemma 1, the result passes $\mathsf{COLM}_0\text{-}\mathsf{Dec}_K(N_1, A_1, \cdot)$ with probability 1 if and only if the modified ciphertext produces the exact same chaining state $W_{\mathrm{mix}}[\ell-1] = W^{(1)}[\ell-1]$ after block $\ell-1$. We now show how an encryption-only, nonce-respecting adversary forces this state collision in two queries.

3.1 Chaining-State Differential Under Block Splicing

Consider an adversary making two forward encryption queries of length ℓ blocks under two *distinct* nonces $N_1 \neq N_2$ (with arbitrary associated data A_1, A_2):

$$\begin{aligned} C^{(1)}[1 \ldots \ell + 1] &\leftarrow \mathsf{COLM}_0\text{-}\mathsf{Enc}_K(N_1, A_1, M^{(1)}), \\ C^{(2)}[1 \ldots \ell + 1] &\leftarrow \mathsf{COLM}_0\text{-}\mathsf{Enc}_K(N_2, A_2, M^{(2)}). \end{aligned} \tag{8}$$

Algorithm 1: COLM_0 Authenticated Encryption (COLM_0-Enc) and Verified Decryption (COLM_0-Dec)

Shared Key & AD Initialization $\mathsf{Init}(K, N, A)$**:**

1. $L \leftarrow E_K(0^{128})$; $A[0] \leftarrow N \,\|\, \mathsf{param}$; $a \leftarrow \lceil |A|/128 \rceil$; if $a \geq 1$, parse A into $A[1], \ldots, A[a]$ (10^*-padded if $|A| < 128a$).

2. Set $\Delta_A[0] \leftarrow 3L$. If $a \geq 1$, set $\Delta_A[j] \leftarrow 3 \cdot 2^j L$ for $1 \leq j < a$, and $\Delta_A[a] \leftarrow 3 \cdot 2^a L$ if $|A| = 128a$ (complete) or $3 \cdot 7 \cdot 2^{a-1} L$ if $|A| < 128a$ (padded).

3. $\mathsf{IV} = W[0] \leftarrow \bigoplus_{j=0}^{a} E_K\big(A[j] \oplus \Delta_A[j]\big)$.

Encryption $\mathsf{COLM}_0\text{-}\mathsf{Enc}_K(N, A, M^*[1 \ldots \ell])$**:** (for complete final block $|M^*[\ell]| = 128$)

1. Run $(L, W[0]) \leftarrow \mathsf{Init}(K, N, A)$ and set the message/tag masks $\Delta_M[1 \ldots \ell+1], \Delta_C[1 \ldots \ell+1]$ via (3).

2. Define the internal message blocks $M[i] \leftarrow M^*[i]$ for $1 \leq i \leq \ell - 1$, and the checksum block:
$M[\ell] \leftarrow \bigoplus_{i=1}^{\ell} M^*[i]$, $M[\ell+1] \leftarrow M[\ell]$.

3. for $i = 1$ **to** $\ell + 1$ **do**
$X[i] \leftarrow E_K\big(M[i] \oplus \Delta_M[i]\big)$;
$Y[i] \leftarrow X[i] \oplus 3W[i-1]$; $W[i] \leftarrow X[i] \oplus 2W[i-1]$; ▷ $(Y[i], W[i]) = \rho(X[i], W[i-1])$
$C[i] \leftarrow E_K(Y[i]) \oplus \Delta_C[i]$;

4. return ciphertext $C[1 \ldots \ell]$ and authentication tag $T = C[\ell+1]$ (total $\ell + 1$ blocks).

Verified Decryption $\mathsf{COLM}_0\text{-}\mathsf{Dec}_K(N, A, C[1 \ldots \ell+1])$**:**

1. Run $(L, W[0]) \leftarrow \mathsf{Init}(K, N, A)$, set $\Delta_M[1 \ldots \ell+1], \Delta_C[1 \ldots \ell+1]$ via (3), and initialize $\mathsf{CS} \leftarrow 0^{128}$.

2. for $i = 1$ **to** ℓ **do**
$Y[i] \leftarrow E_K^{-1}\big(C[i] \oplus \Delta_C[i]\big)$;
$X[i] \leftarrow Y[i] \oplus 3W[i-1]$; $W[i] \leftarrow Y[i] \oplus W[i-1]$; ▷ $(X[i], W[i]) = \rho^{-1}(Y[i], W[i-1])$
$\tilde{M}[i] \leftarrow E_K^{-1}(X[i]) \oplus \Delta_M[i]$;
if $i < \ell$ **then** $\tilde{M}^*[i] \leftarrow \tilde{M}[i]$; $\mathsf{CS} \leftarrow \mathsf{CS} \oplus \tilde{M}^*[i]$;
else $(i = \ell)$: $\tilde{M}^*[\ell] \leftarrow \tilde{M}[\ell] \oplus \mathsf{CS}$; $\mathsf{CS} \leftarrow \mathsf{CS} \oplus \tilde{M}^*[\ell]$; ▷ Note: $\mathsf{CS} = \tilde{M}[\ell]$ by telescoping

3. Tag Check $(i = \ell+1)$**:** Compute $X[\ell+1] \leftarrow E_K(\mathsf{CS} \oplus \Delta_M[\ell+1])$, $Y[\ell+1] \leftarrow X[\ell+1] \oplus 3W[\ell]$, and $\tilde{C}[\ell+1] \leftarrow E_K(Y[\ell+1]) \oplus \Delta_C[\ell+1]$.

4. if $\tilde{C}[\ell+1] = C[\ell+1]$ **then return** $\tilde{M}^*[1 \ldots \ell]$ **else return** $\perp$.

Let $S \subseteq \{1, \ldots, \ell+1\}$ be any splice index set. Define the spliced ciphertext $C^{\mathrm{mix}}[1 \ldots \ell+1]$ by taking $C^{(1)}$ as the base ciphertext and overwriting the blocks indexed by S with those of $C^{(2)}$:

$$C^{\mathrm{mix}}[i] := \begin{cases} C^{(2)}[i] & \text{if } i \in S, \\ C^{(1)}[i] & \text{if } i \notin S, \end{cases} \qquad (1 \leq i \leq \ell+1). \tag{9}$$

When the receiver runs $\mathsf{COLM}_0\text{-}\mathsf{Dec}_K(N_1, A_1, C^{\mathrm{mix}})$, the initial chaining state is $W_{\mathrm{mix}}[0] = \mathsf{IV}_1 = W^{(1)}[0]$. At each block $1 \leq i \leq \ell$, the lower blockcipher decryption produces

$$Y_{\mathrm{mix}}[i] = E_K^{-1}\big(C^{\mathrm{mix}}[i] \oplus \Delta_C[i]\big) = \begin{cases} Y^{(2)}[i] & \text{if } i \in S, \\ Y^{(1)}[i] & \text{if } i \notin S. \end{cases} \tag{10}$$

Define the decryption chaining-state difference relative to Query 1 at step i as $D[i] := W_{\mathrm{mix}}[i] \oplus W^{(1)}[i]$. By the running-XOR identity (6) and $D[0] = 0$, we obtain the exact closed-form expression:

$$D[i] = \bigoplus_{\substack{j \in S \\ j \leq i}} \big(Y^{(1)}[j] \oplus Y^{(2)}[j]\big) \qquad (1 \leq i \leq \ell). \tag{11}$$

Eliminating the Upper-Layer Nonlinearity via Common Plaintext Prefix. In general, $Y^{(1)}[j] \oplus Y^{(2)}[j] = (X^{(1)}[j] \oplus X^{(2)}[j]) \oplus 3(W^{(1)}[j-1] \oplus W^{(2)}[j-1])$. However, suppose the two queried plaintexts agree on their first m blocks: $M^{(1)}[i] = M^{(2)}[i]$ for all $1 \leq i \leq m$ (in the

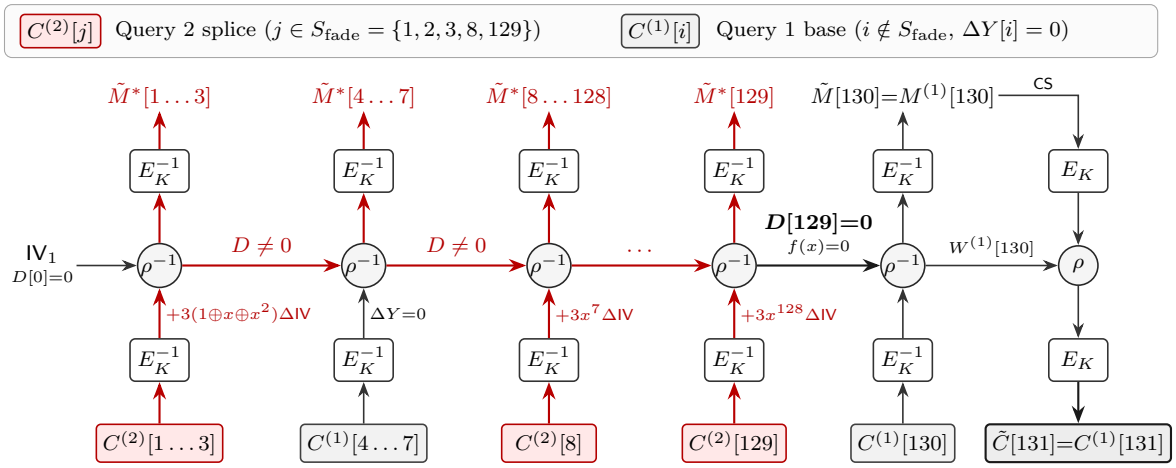

Figure 3: Decryption dataflow of the two-query nonce-respecting Difference-Fade forgery $\mathsf{COLM}_0\text{-}\mathsf{Dec}_K(N_1, A_1, C^{\text{mix}})$ at $\ell = 130$ (Theorem 1; masking XORs omitted for clarity). Splicing $C^{(2)}[j]$ at $S_{\text{fade}} = \{1, 2, 3, 8, 129\}$ injects $3\,\Delta\mathsf{IV}\,(1 \oplus x \oplus x^2 \oplus x^7 \oplus x^{128}) = 3\,\Delta\mathsf{IV} \cdot f(x) = 0$ into the chaining difference $D[129]$, restoring $W_{\text{mix}}[129] = W^{(1)}[129]$ while checksum telescoping (Lemma 1) shields tag verification from all modified plaintext blocks $\tilde{M}^*[1 \dots 129]$.

simplest case, $M^{(1)} = M^{(2)} = M$). Because the input mask $\Delta_M[i] = 2^i L$ in (3) is independent of (N, A), the inputs to the upper blockcipher layer in both queries are identical on blocks $1, \dots, m$:

$$X^{(1)}[i] = E_K\big(M^{(1)}[i] \oplus \Delta_M[i]\big) = E_K\big(M^{(2)}[i] \oplus \Delta_M[i]\big) = X^{(2)}[i] \qquad (1 \le i \le m). \tag{12}$$

Let $\Delta\mathsf{IV} := \mathsf{IV}_1 \oplus \mathsf{IV}_2 = W^{(1)}[0] \oplus W^{(2)}[0]$ denote the initial chaining difference between the two queries. By (4) and (12), the inter-query chaining difference $W^{(1)}[i] \oplus W^{(2)}[i]$ evolves purely linearly across $1 \le i \le m$:

$$W^{(1)}[i] \oplus W^{(2)}[i] = \big(X^{(1)}[i] \oplus X^{(2)}[i]\big) \oplus 2\big(W^{(1)}[i-1] \oplus W^{(2)}[i-1]\big) = 2^i\,\Delta\mathsf{IV}, \tag{13}$$

$$Y^{(1)}[i] \oplus Y^{(2)}[i] = \big(X^{(1)}[i] \oplus X^{(2)}[i]\big) \oplus 3\big(W^{(1)}[i-1] \oplus W^{(2)}[i-1]\big) = 3 \cdot 2^{i-1}\,\Delta\mathsf{IV}. \tag{14}$$

Substituting (14) into (11) and writing $x = 2 \in \mathbb{F}_{2^{128}}$ yields the fundamental polynomial representation of the spliced chaining-state difference:

$$D[i] = 3\,\Delta\mathsf{IV} \cdot \sum_{\substack{j \in S \\ j \le i}} x^{j-1} \quad \text{in } \mathbb{F}_{2^{128}} = \mathbb{F}_2[x]/\big(f(x)\big) \qquad (1 \le i \le m). \tag{15}$$

3.2 Mechanism 1: Difference-Fade Forgery ($\ell \ge 130$)

Our first forgery family—adapting the "difference fades away" concept of Datta et al. [7, Section 6] from INT-RUP decryption queries to two forward encryption queries under distinct nonces—keeps the final ciphertext block $C^{(1)}[\ell]$ and the tag block $C^{(1)}[\ell + 1]$ from Query 1 ($\max(S) \le \ell - 1$). It chooses $S \subseteq \{1, \dots, \ell - 1\}$ so that the polynomial sum in (15) vanishes in $\mathbb{F}_{2^{128}}$. Figure 3 depicts the resulting decryption dataflow at the minimal length $\ell = 130$.

Theorem 1 (Two-Query Nonce-Respecting Difference-Fade Forgery on COLM_0)**.** *Let $\ell \ge 130$ be any integer, let M be any block-aligned plaintext of ℓ blocks (16ℓ bytes), and let $(N_1, A_1), (N_2, A_2)$ be any two pairs with $N_1 \ne N_2$ (nonce-respecting). Given the two ciphertexts $C^{(1)} \leftarrow \mathsf{COLM}_0\text{-}\mathsf{Enc}_K(N_1, A_1, M)$ and $C^{(2)} \leftarrow \mathsf{COLM}_0\text{-}\mathsf{Enc}_K(N_2, A_2, M)$, define the 5-element splice index set*

$$S_{\text{fade}} := \operatorname{supp}\big(f(x)\big) + 1 = \{1, 2, 3, 8, 129\}. \tag{16}$$

Then the spliced ciphertext C^{mix} defined by (9) with $S = S_{\mathrm{fade}}$ is accepted by the verification routine $\mathsf{COLM}_0\text{-}\mathsf{Dec}_K(N_1, A_1, C^{\mathrm{mix}})$ with probability 1 (provided $\mathsf{IV}_1 \neq \mathsf{IV}_2$, which holds deterministically for every key K whenever $A_1 = A_2$ by the bijectivity of E_K). Moreover, $\ell_{\min} = 130$ is the strict minimum block length for which any non-empty fade splice exists.

Proof. Since $M^{(1)} = M^{(2)} = M$ and $\max(S_{\mathrm{fade}}) = 129 \leq \ell - 1$, the internal message blocks $M^{(1)}[i]$ and $M^{(2)}[i]$ coincide for all $1 \leq i \leq \ell$. By (15), at every block $i \in \{129, \ldots, \ell - 1\}$, the chaining-state difference is

$$D[i] = 3\,\Delta\mathsf{IV} \cdot \sum_{j \in \{1,2,3,8,129\}} x^{j-1} = 3\,\Delta\mathsf{IV} \cdot \left(1 \oplus x \oplus x^2 \oplus x^7 \oplus x^{128}\right) = 3\,\Delta\mathsf{IV} \cdot f(x) = 0 \quad \text{in } \mathbb{F}_{2^{128}}. \tag{17}$$

In particular, $D[\ell - 1] = 0$, so $W_{\mathrm{mix}}[\ell - 1] = W^{(1)}[\ell - 1]$. Because $C^{\mathrm{mix}}[\ell] = C^{(1)}[\ell]$ and $C^{\mathrm{mix}}[\ell + 1] = C^{(1)}[\ell + 1]$ ($129 < \ell < \ell + 1$), Lemma 1 implies that $\tilde{M}_{\mathrm{mix}}[\ell] = M^{(1)}[\ell]$ and $W_{\mathrm{mix}}[\ell] = W^{(1)}[\ell]$. The recomputed tag block $\tilde{C}^{\mathrm{mix}}[\ell+1]$ therefore equals $C^{(1)}[\ell+1] = C^{\mathrm{mix}}[\ell+1]$ identically. Whenever $\Delta\mathsf{IV} \neq 0$, $Y^{(1)}[j] \neq Y^{(2)}[j]$ for all $j \in S_{\mathrm{fade}}$, so $C^{\mathrm{mix}} \neq C^{(1)}$ and $C^{\mathrm{mix}} \neq C^{(2)}$ is a valid, non-trivial $\mathsf{INT\text{-}CTXT}$ forgery.

To see that $\ell_{\min} = 130$ is tight, note that retaining the tag block $C^{\mathrm{mix}}[\ell + 1] = C^{(1)}[\ell + 1]$ requires both $\tilde{M}_{\mathrm{mix}}[\ell] = M^{(1)}[\ell]$ (hence $X_{\mathrm{mix}}[\ell] = X^{(1)}[\ell]$) and $W_{\mathrm{mix}}[\ell] = W^{(1)}[\ell]$. Since ρ^{-1} is a bijection, this forces $Y_{\mathrm{mix}}[\ell] = Y^{(1)}[\ell]$ and $W_{\mathrm{mix}}[\ell - 1] = W^{(1)}[\ell - 1]$ ($D[\ell - 1] = 0$), so index ℓ cannot belong to S. Thus $S \subseteq \{1, \ldots, \ell - 1\}$, and $\sum_{j \in S} x^{j-1} = 0$ requires a non-trivial $\mathbb{F}_2$-linear dependency among $\{x^0, x^1, \ldots, x^{\ell-2}\}$. Because $\deg(f) = 128$, the powers $\{1, x, \ldots, x^{127}\}$ form a basis of $\mathbb{F}_{2^{128}}$ over $\mathbb{F}_2$, so $\mathrm{rank}_{\mathbb{F}_2}\{x^0, \ldots, x^{\ell-2}\} = \ell - 1$ for all $\ell \leq 129$. The first dependency therefore occurs at $\ell - 2 = 128$, i.e., $\ell_{\min} = 130$. □

3.3 Mechanism 2: State-Transfer Forgery ($\ell \geq 129$)

Instead of fading $D[\ell - 1]$ back to zero and reusing Query 1's tag block, our second forgery family drives the spliced chaining state W_{mix} onto *Query 2's chaining state* $W^{(2)}$ and transplants Query 2's final block and tag block (generalizing the fixed-IV intermediate-tag splice of Datta and Nandi [8, Section 5.2] to distinct nonces and full-message checksum tags on COLM_0).

Theorem 2 (Two-Query Nonce-Respecting State-Transfer and Tag-Transplant Forgery on COLM_0)**.** *Let $\ell \geq 129$ be any integer, let $m \in \{129, \ldots, \ell\}$ be any run-start index, and let $C^{(1)}, C^{(2)}$ be the ciphertexts of the same ℓ-block message M under distinct nonces $N_1 \neq N_2$. Define the splice index set*

$$S_{\mathrm{trans}}(\ell, m) := \left(\mathrm{supp}\left(3^{-1}x^{m-1}\right) + 1\right) \cup \{m, m + 1, \ldots, \ell + 1\}. \tag{18}$$

At the minimal length $\ell = m = 129$ blocks (2,064 bytes), $S_{\mathrm{trans}}(129, 129)$ is the 8-element set

$$S_{\mathrm{trans}}(129, 129) = \{1, 3, 4, 5, 6, 7, 129, 130\}. \tag{19}$$

Then C^{mix} spliced on $S = S_{\mathrm{trans}}(\ell, m)$ is accepted by $\mathsf{COLM}_0\text{-}\mathsf{Dec}_K(N_1, A_1, C^{\mathrm{mix}})$ with probability 1 (provided $\mathsf{IV}_1 \neq \mathsf{IV}_2$, which holds deterministically whenever $A_1 = A_2$), and $\ell_{\min} = 129$ is the strict minimum block length of this family.

Proof. Observe first that splicing block i ($i \in S$, so $Y_{\mathrm{mix}}[i] = Y^{(2)}[i]$) makes the decrypted upper-layer state $X_{\mathrm{mix}}[i] = Y^{(2)}[i] \oplus 3W_{\mathrm{mix}}[i-1]$ and updated chaining state $W_{\mathrm{mix}}[i] = Y^{(2)}[i] \oplus W_{\mathrm{mix}}[i-1]$ equal to Query 2's states $(X^{(2)}[i], W^{(2)}[i])$. This holds if and only if $W_{\mathrm{mix}}[i - 1] = W^{(2)}[i - 1]$. Once $W_{\mathrm{mix}}[m - 1] = W^{(2)}[m - 1]$ holds at index $m - 1$, splicing every subsequent block $i \in \{m, m + 1, \ldots, \ell + 1\}$ inductively maintains $W_{\mathrm{mix}}[i] = W^{(2)}[i]$ and $\tilde{M}_{\mathrm{mix}}[\ell] = M^{(2)}[\ell]$. The transplanted tag block $C^{\mathrm{mix}}[\ell+1] = C^{(2)}[\ell+1]$ therefore verifies under (N_1, A_1) with probability 1.

By (13) and (15), the state-transfer condition $W_{\text{mix}}[m-1] = W^{(2)}[m-1]$ is equivalent to $D[m-1] = W^{(1)}[m-1] \oplus W^{(2)}[m-1] = x^{m-1}\Delta\mathsf{IV}$, which holds for all $\Delta\mathsf{IV}$ if and only if

$$3 \sum_{\substack{j \in S \\ j \le m-1}} x^{j-1} = x^{m-1} \quad \Longleftrightarrow \quad \sum_{\substack{j \in S \\ j \le m-1}} x^{j-1} = 3^{-1} x^{m-1} \quad \text{in } \mathbb{F}_{2^{128}}. \tag{20}$$

Since $S \cap \{1, \ldots, m-1\}$ can only use exponents $\{0, 1, \ldots, m-2\}$, equation (20) admits a solution if and only if

$$\deg\left(3^{-1} x^{m-1} \bmod f(x)\right) \le m-2 \qquad \text{(Condition C)}. \tag{21}$$

In $\mathbb{F}_{2^{128}} = \mathbb{F}_2[x]/(x^{128} + x^7 + x^2 + x + 1)$, the element $(x+1)^{-1} = x \oplus \sum_{k=7}^{127} x^k$ has degree 127. Direct computation shows that $\deg(3^{-1}x^{m-1} \bmod f) = 127 > m-2$ for every $1 \le m \le 128$. At $m = 129$, $3^{-1}x^{128} = 3^{-1}(x^7 \oplus x^2 \oplus x \oplus 1) = x^6 \oplus x^5 \oplus x^4 \oplus x^3 \oplus x^2 \oplus 1$, which has degree $6 \le 127$ and support $\{0, 2, 3, 4, 5, 6\}$. Shifting indices by $+1$ gives the prefix $\{1, 3, 4, 5, 6, 7\}$, and adjoining the contiguous suffix $\{129, \ldots, \ell + 1\}$ yields (18)–(19). □

3.4 Extensions and Comparison with Nonce-Misuse Splicing

The algebraic framework of (11)–(15) yields three additional observations:

1. **Common-Prefix Relaxation:** In Theorem 1, the two queried messages $M^{(1)}$ (of length $\ell_1 \ge 130$ blocks) and $M^{(2)}$ (of any length $\ell_2 \ge 130$ blocks, including messages with an incomplete final block) need *not* be identical. They only need to agree on their first 129 blocks ($M^{(1)}[1 \ldots 129] = M^{(2)}[1 \ldots 129]$, a 2,064-byte common prefix). Because $\max(S_{\text{fade}}) = 129$, decryption of C^{mix} never accesses $C^{(2)}[i]$ for $i \ge 130$.

2. **Same-Nonce / Distinct-AD Relaxation:** Because $W[0] = \mathsf{IV}$ is a function of (N, A), the condition $\Delta\mathsf{IV} \neq 0$ in Theorems 1 and 2 is satisfied whenever the two queries use distinct nonces under arbitrary associated data (including empty or non-block-aligned AD). It is equally satisfied when they repeat a nonce under distinct associated data $A_1 \neq A_2$.

3. **Incompatibility with First-Block Message Differences:** If two queries also differ in the first plaintext block ($M^{(1)}[1] \neq M^{(2)}[1]$, inducing an upper-layer difference $\delta = X^{(1)}[1] \oplus X^{(2)}[1] \neq 0$) while agreeing on blocks $2, \ldots, \ell$, then splicing requires $\Delta\mathsf{IV} = 0$ (the nonce-reuse setting of [8, 7]):

Proposition 1 (Two-Query Splicing with a First-Block Message Difference)**.** *Let Query 1 and Query 2 agree on plaintext blocks* $2, \ldots, \ell$ *(*$\ell \ge 130$*), with initial chaining difference* $\Delta\mathsf{IV} = \mathsf{IV}_1 \oplus \mathsf{IV}_2$ *and block-1 upper-layer difference* $\delta = X^{(1)}[1] \oplus X^{(2)}[1]$*. Then for any splice set* $S \subseteq \{1, \ldots, 129\}$*, the chaining difference at block* 129 *is*

$$D[129] = \Big(\mathbf{1}_{[1 \in S]} \oplus 3u\Big)\delta \;\oplus\; 3\Big(\mathbf{1}_{[1 \in S]} \oplus 2u\Big)\Delta\mathsf{IV}, \qquad \textit{where } u = \sum_{\substack{j \in S \\ j \ge 2}} x^{j-2} \in \mathbb{F}_{2^{128}}. \tag{22}$$

In particular, when $\Delta\mathsf{IV} = 0$ *and* $\delta \neq 0$ *(nonce reuse),* $D[129] = 0$ *holds if and only if* $1 \in S$ *and* $u = 3^{-1}$*, recovering the* 123*-element* 3^{-1}*-expansion splice set* $S_B = \{1\} \cup \{i + 2 : \mathsf{bit}_i(3^{-1}) = 1\}$ *of Datta et al. [7, Lemma 2] (or the* 5*-element shifted set* $S_{\text{fade}} + 1 = \{2, 3, 4, 9, 130\}$ *at* $\ell \ge 131$*). Conversely, when both* $\Delta\mathsf{IV} \neq 0$ *and* $\delta \neq 0$*, canceling the two independent pseudorandom values* δ *and* $\Delta\mathsf{IV}$ *requires* $\mathbf{1}_{[1 \in S]} \oplus 3u = 0$ *and* $\mathbf{1}_{[1 \in S]} \oplus 2u = 0$*, which forces* $u = 0$*,* $\mathbf{1}_{[1 \in S]} = 0$*, and hence* $S = \emptyset$*.*

4 Analysis of the Official Security Proofs

Two security proofs have been published by the COLM designers for COLM_0 ($\tau = 0$):

1. **The 2024 *Journal of Cryptology* Proof [5, Section 4, pp. 15–25]:** A full Patarin H-coefficient proof for unified online authenticated encryption (oAE) security (Theorem 2, Lemma 1, Lemma 2, and Section 4.4).

2. **The 2016 CAESAR Round-3 Addendum [4, Section 2, pp. 2–4]:** An H-coefficient proof bounding the forgery advantage $\mathsf{Adv}^{\mathrm{auth}}_{\mathsf{COLM}_\Pi}$ (Theorem 2, Lemma 3, and Lemma 4).

We evaluate both theorems at the parameters of our two-query adversary ($q = 2$ forward encryption queries, $s = 1$ forgery attempt, and zero prior decryption/RUP queries). We then identify the exact mathematical steps at which each proof breaks down against this two-query adversary.

4.1 The 2024 *Journal of Cryptology* Security Proof

Theorem 2 of [5, Section 4, p. 15] states verbatim:

> "***Theorem 2.*** *Let us consider an adversary $\mathcal{A}$ that makes q encryption queries and s forging attempts at an overall aggregate of total a associated data and m message blocks and tries to distinguish COLM_0 from a random online function. Let $\sigma = a + 2m + 3(q + s)$. The* oAE *advantage of COLM_0 is given by*
>
> $$\mathsf{Adv}^{\mathsf{oAE}}_{\mathsf{COLM}_0}(\mathcal{A}) \le \mathsf{Adv}^{\mathsf{SPRP}}_{E}(\mathcal{A}') + \frac{7\sigma^2 + 3\sigma + qs + 3s + 2s\sigma}{2^n},\text{"} \tag{23}$$

where Lemma 2 [5, p. 23] bounds the probability of bad transcripts by $\epsilon_{\mathrm{bad}} \le (7\sigma^2 + 3\sigma + qs)/2^n$, and Section 4.4 [5, pp. 24–25] bounds the good-transcript forgery ratio by $\epsilon_{\mathrm{ratio}} \le (3s + 2s\sigma)/2^n$.

At our adversary's parameters ($n = 128$, $q = 2$, $s = 1$, $a_i \le 2$ giving $a \le 6$, and $\ell_i \in \{129, 130\}$ giving $m \le 390$), we have $\sigma = a + 2m + 3(q + s) \le 6 + 780 + 9 = 795$. Substituting $\sigma = 795$ into (23) gives the claimed information-theoretic bound

$$\epsilon_{\mathrm{bad}} + \epsilon_{\mathrm{ratio}} = \frac{7(795)^2 + 3(795) + 2 + 3 + 2(795)}{2^{128}} = \frac{4{,}428{,}155}{2^{128}} \approx 2^{-105.92}, \tag{24}$$

whereas our adversary achieves $\mathsf{Adv}^{\mathsf{oAE}}_{\mathsf{COLM}_0}(\mathcal{A}) = 1$, exceeding the claimed bound by a factor of $2^{105.92}$.

Flaw 1: Checksum Direction Inversion in Event BadSamp. In the 2024 proof, Bad Event B6 [5, p. 22] is defined as the event that a non-trivial decryption query $i \in \mathcal{D}$ has *no* fresh upper-layer output block $X_i^-[j]$ across all $1 \le j \le \ell_i$:

$$\mathsf{B6} \;:\; \exists i \in \mathcal{D} \;\text{ s.t. }\; X_i^-[j] = X_{i_j}^+[j] \quad \forall j \le \ell_i. \tag{25}$$

In our Difference-Fade forgery (Theorem 1, $\ell = 130$), the spliced blocks $S_{\mathrm{fade}} = \{1, 2, 3, 8, 129\}$ alter the intermediate chaining states $W_{\mathrm{mix}}[j]$ for $1 \le j \le 128$. Hence, $X_i^-[j] = Y^{(2)}[j] \oplus 3W_{\mathrm{mix}}[j-1]$ is genuinely *fresh* for $j \in \{1, \ldots, 128\}$. Bad Event B6 therefore does not occur, and the transcript satisfies the good-transcript definition of Section 4.3.

Next, in Section 4.4 (*Calculation of* $\epsilon_{\mathrm{ratio}}$ [5, p. 24]), the authors bound the conditional forgery probability $\epsilon_i = \Pr[\mathsf{Forge}_i \mid \tau_e, \tau_d]$ on a good transcript. They argue verbatim:

> *"By the properties of a good transcript, the set $\{j : X_i^-[j]$ is fresh$\}$ is nonempty. Let j_i denote the smallest such index. Now we sample all $MM_i^-[k]$ values randomly in without replacement manner from $\{0,1\}^n \setminus \mathcal{O}$, where $k \in \{j : X_i^-[j]$ is fresh$\}$. All other $MM_i^-[j]$ values are set... Now we say that the sampling of MM_i^- values is bad, denoted by the event* BadSamp*, if $MM_i^-[\ell_i]$ is not fresh, i.e., $MM_i^-[\ell_i] \in \mathcal{I} \cup \{MM_i^-[j] : j < \ell_i\}$. It is easy to see that the probability of the event* BadSamp *is bounded by $1/(2^n - 2\sigma)$* due to the randomness of $MM_i^-[j_i]$ [emphasis added]. *Now conditioned on the event that* BadSamp *does not occur, we can sample distinct values of $CC_i^-[\ell_i + 1]$ uniformly at random..."*

This step contains a direction-inversion error between encryption and decryption:

- The proof implicitly assumes that in a decryption query $i \in \mathcal{D}$, the ℓ_i-th internal block $MM_i^-[\ell_i]$ (and thus the (ℓ_i+1)-th input $M_i^-[\ell_i+1] = M_i^-[\ell_i]$ governing the tag $CC_i^-[\ell_i+1]$) incorporates the XOR sum of the preceding decrypted blocks $M_i^-[1] \oplus \cdots \oplus M_i^-[\ell_i - 1]$. That sum would inherit the randomness of $MM_i^-[j_i]$ ($j_i = 1 < \ell_i$).
- In fact, according to the decryption specification $\mathsf{COLM}_0.\mathsf{dec}$ [5, Figure 3, p. 12], $MM_i^-[\ell_i] = E_K^{-1}(X_i^-[\ell_i])$ and $M_i^-[\ell_i] = MM_i^-[\ell_i] \oplus \Delta_2[\ell_i]$ (denoted $\Delta_M[\ell_i]$ in Section 2) are determined *solely* by the ℓ_i-th ciphertext block $C_i^-[\ell_i]$ and the incoming chaining state $W_i^-[\ell_i - 1]$ via $X_i^-[\ell_i] = Y_i^-[\ell_i] \oplus 3W_i^-[\ell_i - 1]$. The preceding blocks $M_i^-[1 \ldots \ell_i - 1]$ are XORed only into the external output plaintext block $M_i^\star[\ell_i] \leftarrow M_i^-[1] \oplus \cdots \oplus M_i^-[\ell_i]$. The tag-generation input, by contrast, is set to $M_i^-[\ell_i + 1] \leftarrow M_i^-[\ell_i]$ rather than $M_i^\star[\ell_i]$.
- In our Difference-Fade forgery ($\ell_i = 130$), $D[129] = 0$ ensures $W_i^-[129] = W_1^+[129]$, while $C_i^-[130] = C_1^+[130]$ ensures $Y_i^-[130] = Y_1^+[130]$. As a result, $X_i^-[130] = X_1^+[130]$ is *not fresh*, forcing $MM_i^-[130] = MM_1^+[130]$ and $CC_i^-[131] = CC_1^+[131]$ with probability 1, independently of $MM_i^-[j_i]$.

Flaw 2: Overlooking $\mathbb{F}_{2^{128}}$ Polynomial Dependencies in Lemma 1. In our State-Transfer forgery (Theorem 2, $\ell_i = 129$) and in the second subcase of Bad Event B6 ("$\forall j, CC_i^-[j]$ is not fresh" [5, p. 23]), every ciphertext block of C_i^- is copied from either Query 1 or Query 2 at the corresponding block index j. To argue that $L_X(W[0], Y[1\,..\,j+1])$ cannot coincide with $X'[j+1]$, the authors invoke Lemma 1 [5, p. 20], expanding $L_X(W[0], Y[1\,..\,j+1])$ as

$$L_X(W[0], Y[1\,..\,j{+}1]) = X'[j{+}1] \oplus 3(X[j] \oplus X'[j]) \oplus \cdots \oplus 3{\cdot}2^{j-1}(X'[1] \oplus X[1]) \oplus 3{\cdot}2^j(W'[0] \oplus W[0]), \tag{26}$$

and concluding that equality with $X'[j+1]$ forces $W'[0] \equiv W[0]$ and $X'[r] \equiv X[r]$ for all r. This implication fails for $\ell_i \geq 129$. When two queries share a 129-block plaintext prefix, $X_1^+[r] \equiv X_2^+[r]$ already holds for $1 \leq r \leq 128$. The accumulated initial-value terms $\Delta W[0] = W_1^+[0] \oplus W_2^+[0]$ across S then sum to $\big(3\sum_{r \in S, r \leq 128} 2^{r-1} \oplus 2^{128}\big)\Delta W[0]$. That sum vanishes identically in $\mathbb{F}_{2^{128}}$ whenever $S \cap \{1, \ldots, 128\} = \mathrm{supp}(3^{-1}2^{128}) + 1 = \{1, 3, 4, 5, 6, 7\}$, despite $W_1^+[0] \neq W_2^+[0]$.

4.2 The 2016 CAESAR Round-3 Addendum Proof

In the earlier 2016 CAESAR Round-3 addendum [4, Section 2, p. 3], Theorem 2 claims that for $\sigma = \sum_{i=1}^{q+s}(a_i + \ell_i)$ and $\sigma_{\mathrm{auth}} = \sigma + q$,

$$\mathsf{Adv}^{\mathrm{auth}}_{\mathsf{COLM}_\Pi}(\mathcal{A}) \leq \frac{9\sigma_{\mathrm{auth}}^2}{2^n} + \frac{s}{2^n}. \tag{27}$$

At $q = 2, s = 1, a_i = 2, \ell_i = 130$ ($\sigma = 396, \sigma_{\mathrm{auth}} = 398$), (27) evaluates to $2^{-107.56}$. In that proof, Step I (Condition 2 [4, p. 3]) defines a "good online forge view" (τ_{good}) by requiring that for every forgery attempt $j > q$, the submitted tag block $C_j[\ell_j+1]$ be *fresh* ("distinct and different from all

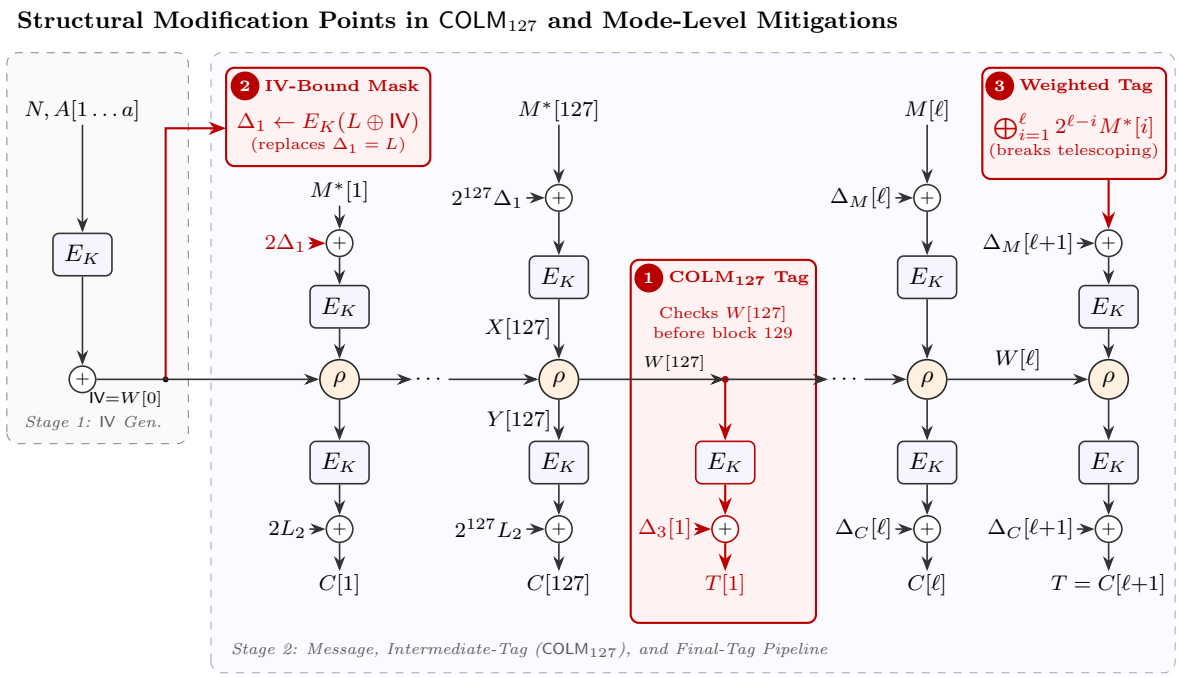

Figure 4: Overview of the COLM pipeline showing the three structural modification points that prevent two-query splicing (compare Figure 1(a)): **(1)** Intermediate tag $T[1] = E_K(W[127]) \oplus \Delta_3[1]$ in COLM_{127} ($\tau = 127$), which verifies $W[127]$ before the degree-128 field relation at block 129 can occur; **(2)** IV-bound upper-layer mask base $\Delta_1 \leftarrow E_K(L \oplus \mathsf{IV})$, which couples the input masks to (N, A) and prevents cross-query $X[i]$ cancellation; and **(3)** position-weighted tag checksum $M[\ell+1] \leftarrow \bigoplus_{i=1}^{\ell} 2^{\ell-i} M^*[i]$, which prevents decryption checksum telescoping.

other $C_i[j']$'s"). Lemma 3 then bounds $\Pr[\tau \notin \tau_{\text{good}}] \le \frac{(q+\sum_{i=1}^{q}\ell_i)^2}{2^{n+1}} + \frac{s(q+s+\sigma)}{2^n} = \frac{34,721}{2^{128}} \approx 2^{-112.92}$ (further upper-bounded by $2\sigma_{\text{auth}}^2/2^n \approx 2^{-109.73}$) in the ideal world, while Lemma 4 [4, pp. 3–4] lower-bounds the real-world interpolation ratio by claiming that choosing fresh intermediate blocks MM yields fresh checksums and fresh final Y blocks. This argument fails for the same two reasons identified above: in COLM_0, checksum telescoping decouples the tag input $M_3[\ell+1] = M_3[\ell]$ from intermediate blocks $MM_3[1 \ldots \ell-1]$, and aligning $W_{\text{mix}}[\ell-1]$ with $W^{(1)}[\ell-1]$ (Theorem 1) or $W^{(2)}[\ell-1]$ (Theorem 2) allows the adversary to reuse $C_3[\ell+1] := C_1[\ell+1]$ or $C_2[\ell+1]$ deterministically.

5 Resistance of COLM_{127} and Mode-Level Mitigations

Recall from Section 3 that our two-query nonce-respecting forgeries on COLM_0 rely on the simultaneous presence of three structural properties: (i) an uninterrupted chaining horizon of at least 129 message blocks, allowing the degree-128 field polynomial $f(x)$ to cancel $\Delta\mathsf{IV}$; (ii) a nonce-independent upper-layer mask base $\Delta_1 = L$, allowing $X^{(1)}[i] = X^{(2)}[i]$ to cancel across distinct nonces; and (iii) an unweighted duplicate checksum $M[\ell+1] = M[\ell] = \bigoplus_{i=1}^{\ell} M^*[i]$, allowing intermediate plaintext corruptions to telescope out during decryption (Lemma 1). As illustrated in Figure 4, breaking any one of these three links prevents two-query splicing. We first analyze how COLM_{127} ($\tau = 127$, **Modification 1**) truncates the chaining horizon one block before the degree-128 threshold, and then evaluate two zero-expansion mode-level modifications (**Modifications 2 and 3**).

5.1 One-Block Degree Obstruction in COLM_{127}

As shown in **Modification 1** of Figure 4, COLM_τ ($\tau \geq 1$) emits an intermediate authentication tag $T[j] = E_K(W[\tau j]) \oplus \Delta_3[j_{\text{tag}}]$ after every τ message blocks ($i = \tau j$ for $j = 1, 2, \ldots, \lfloor (\ell - 1)/\tau \rfloor$). During decryption, the receiver verifies $T[j]$ against $W_{\text{mix}}[\tau j]$ before processing subsequent blocks [3, Sections 1.4 and 1.5.4, pp. 6, 8]. In ELmE [8, Lemma 9 and Section 5.2] and COLM [7, Remark 1, p. 155], the stride $\tau = n - 1 = 127$ was chosen specifically so that any single-segment message-difference splice lacks the $n = 128$ blocks needed to invoke $f(x) = 0$. The following proposition shows that this one-block degree barrier also precludes both cross-nonce ΔIV splicing families:

Proposition 2 (Two-Query Splice Obstruction for COLM_τ at $\tau \leq 127$)**.** *For every intermediate-tag stride $1 \leq \tau \leq 127$, no non-empty two-query splice on blocks $\{1, \ldots, \tau\}$ can satisfy the intermediate-tag check at block τ, regardless of whether the intermediate tag $T^{\text{mix}}[1]$ is retained from Query 1 ($T^{(1)}[1]$) or transplanted from Query 2 ($T^{(2)}[1]$). The safety margin of COLM_{127} ($\tau = 127$) against two-query splicing is* exactly one block *($\tau^* = 128$).*

Proof. Consider the first τ-block segment $\{1, \ldots, \tau\}$, and let $S_1 = S \cap \{1, \ldots, \tau\}$ denote the spliced positions within this segment. Because both queries have the same intermediate-tag index, the mask $\Delta_3[1]$ cancels, so deterministic acceptance at the first intermediate tag requires $W_{\text{mix}}[\tau]$ to match either $W^{(1)}[\tau]$ or $W^{(2)}[\tau]$:

1. **Retaining $T^{\text{mix}}[1] = T^{(1)}[1]$ (Fade within the first segment):** Requires $W_{\text{mix}}[\tau] = W^{(1)}[\tau]$, i.e., $\sum_{j \in S_1} x^{j-1} = 0$ in $\mathbb{F}_{2^{128}}$. Since $S_1 \subseteq \{1, \ldots, \tau\}$, the exponents $j - 1$ lie in $\{0, \ldots, \tau - 1\} \subseteq \{0, \ldots, 126\}$. Because $\{1, x, \ldots, x^{127}\}$ is linearly independent over $\mathbb{F}_2$, $\sum_{j \in S_1} x^{j-1} = 0$ forces $S_1 = \emptyset$.

2. **Transplanting $T^{\text{mix}}[1] = T^{(2)}[1]$ (State transfer at block τ):** Requires $W_{\text{mix}}[\tau] = W^{(2)}[\tau]$, i.e., $D[\tau] = W^{(1)}[\tau] \oplus W^{(2)}[\tau] = x^\tau \Delta\text{IV}$. By (15), this is equivalent to $\sum_{j \in S_1} x^{j-1} = 3^{-1} x^\tau \bmod f(x)$, which admits a subset $S_1 \subseteq \{1, \ldots, \tau\}$ if and only if $\deg(3^{-1} x^\tau \bmod f(x)) \leq \tau - 1$. However, since $(x + 1) \cdot (3^{-1} x^\tau) \equiv x^\tau \pmod{f(x)}$ and $x + 1 \nmid x^\tau$ in $\mathbb{F}_2[x]$ for $1 \leq \tau \leq 127$, the unique polynomial of degree at most 127 whose product with $(x + 1)$ equals $x^\tau + f(x)$ has leading term x^{127}, yielding

$$\deg\left(3^{-1} x^\tau \bmod f(x)\right) = 127 > \tau - 1 \qquad \forall\, 1 \leq \tau \leq 127. \tag{28}$$

   By contrast, at $\tau = 128$, $\deg(3^{-1} x^{128} \bmod f) = 6 \leq 127$. A hypothetical COLM_{128} instance would therefore admit the 6-block state-transfer splice $S_1 = \{1, 3, 4, 5, 6, 7\}$ with $T^{(2)}[1]$ transplanted at block 128.

Thus $\tau = 127$ is the maximum stride that precludes two-query splicing within the initial segment. □

5.2 Mitigation Options and Theoretical Overheads

Corresponding to **Modifications 1–3** in Figure 4, Table 2 compares the three mitigations and their overheads:

1. **Modification 1 — Adopting COLM_{127} ($\tau = 127$, Recommended Mode):** Deploying COLM_{127} requires no change to the COLM specification, incurring an asymptotic ciphertext expansion of 16 bytes per 127 blocks ($1/127 \approx +0.79\%$) and one additional blockcipher invocation per 127 blocks ($1/254 \approx +0.39\%$).

2. **Modification 2 — IV-Bound Input Masking ($\Delta_1 \leftarrow E_K(L \oplus \text{IV})$):** Replacing the nonce-independent message mask base $\Delta_1 = L$ in (2) with an IV-dependent base $\Delta_1 = E_K(L \oplus \text{IV})$

Table 2: Comparison of mode-level mitigations for COLM against two-query splicing and their overheads.

Scheme / Mitigation	Ciphertext Size[a]	AES Calls[a]	Unique Nonce	Repeated Nonce
Baseline COLM_0	2,096 B (+0 B)	266 (+0)	Vulnerable	Vulnerable
Intermediate Tags (COLM_{127})[b]	2,112 B (+16 B)	267 (+1)	**Resists Splice**	**Resists Splice**
IV-Bound Input Mask[c]	2,096 B (+0 B)	267 (+1)	**Resists Splice**	Vulnerable
Weighted Tag Checksum[d]	2,096 B (+0 B)	266 (+0)	**Resists Splice**	**Resists Splice**

[a]Evaluated on a benchmark message of $\ell = 130$ blocks (2,080 B) and $|A| = 32$ B (2 blocks); parenthesized values denote overhead relative to COLM_0.
[b]Recommended parameter set with intermediate tag stride $\tau = 127$ (Proposition 2).
[c]Replaces $\Delta_1 \leftarrow L$ with $\Delta_1 \leftarrow E_K(L \oplus \mathsf{IV})$ after Stage 1; remains vulnerable if both nonce and AD repeat (Proposition 1).
[d]Replaces $M[\ell+1] = M[\ell]$ with the polynomial hash $M[\ell+1] \leftarrow \bigoplus_{i=1}^{\ell} 2^{\ell-i} M^*[i]$, preventing checksum telescoping (Lemma 1). Establishing full oAE security up to the birthday bound requires a complete re-proof of the BadSamp and B6 bounds in [5].

Table 3: Experimental verification of our two-query nonce-respecting forgeries on COLM_0.

Attack Family	Splice Set S	Blocks (ℓ)	CAESAR Ref.	AES-NI	JoC 2024[c]
Difference Fade[a]	$\{1, 2, 3, 8, 129\}$	130	**1000/1000**	**1000/1000**	**1000/1000**
State Transfer[b]	$\{1, 3, 4, 5, 6, 7, 129, 130\}$	129	**1000/1000**	**1000/1000**	**1000/1000**

[a]Theorem 1. [b]Theorem 2. [c]Evaluated on both $\mathsf{AES\text{-}COLM}_0$-128 and $\mathsf{AES\text{-}COLM}_0$-256 [5] using 2 encryption queries and 0 decryption queries under distinct nonces $N_1 \neq N_2$.

after Stage 1 ensures independent pseudorandom input masks whenever $\mathsf{IV}_1 \neq \mathsf{IV}_2$, preventing the upper-layer cancellation (12) across distinct nonces or distinct AD. This introduces zero ciphertext expansion and a single additional blockcipher call per message (+0.38% at $\ell = 130$), though it does not eliminate decryption checksum telescoping (Lemma 1) or prevent repeated-(nonce, AD) misuse splices (Proposition 1).

3. **Modification 3 — Position-Weighted Tag Checksum ($M[\ell+1] \leftarrow \bigoplus_{i=1}^{\ell} 2^{\ell-i} M^*[i]$):** Replacing the unweighted tag checksum $M[\ell+1] = M[\ell] = \bigoplus_{i=1}^{\ell} M^*[i]$ with the Horner-evaluated polynomial hash $S_\ell = \bigoplus_{i=1}^{\ell} 2^{\ell-i} M^*[i]$ (while preserving $M[\ell] = \bigoplus_{i=1}^{\ell} M^*[i]$ at block ℓ for single-pass plaintext recovery) eliminates Lemma 1 and restores the intermediate-block dependency assumed by the BadSamp step in Section 4.4 of [5] (though a complete security proof would also require re-bounding event B6 in Lemma 1 of [5]). Any splice altering plaintext blocks $1, \ldots, \ell-1$ induces pseudorandom decryption differences $\Delta M^*[j] = E_K^{-1}(X_{\mathrm{mix}}[j]) \oplus E_K^{-1}(X^{(1)}[j])$, perturbing the tag input by $\bigoplus_{j=1}^{\ell-1}(2^{\ell-j} \oplus 1)\Delta M^*[j] \neq 0$ with zero ciphertext expansion and zero additional blockcipher calls (requiring only one $\mathbb{F}_{2^{128}}$ doubling per block).

6 Experimental Verification

To confirm the practical feasibility of our attacks, we verified both the Difference-Fade forgery (Theorem 1) and the State-Transfer forgery (Theorem 2) against the official implementations of COLM_0.

As summarized in Table 3, in each trial the test harness samples a fresh random key K, two distinct nonces $N_1 \neq N_2$, associated data A_1, A_2, and a random plaintext M, queries the encryption oracle twice, splices the resulting ciphertexts at the prescribed block indices S, and submits C^{mix} to the verified decryption routine $\mathsf{COLM}_0\text{-}\mathsf{Dec}_K(N_1, A_1, C^{\mathrm{mix}})$. Across 1,000 independent random trials per attack family, both constructions succeeded with a 100%

acceptance rate across all tested implementations and both key sizes, confirming that the algebraic state collision and checksum telescoping hold identically in practice.

Use of AI. The cryptanalytic results and artifacts were initially developed through an autonomous agent system powered by a MiniMax internal model. We validated the key technical claims and refined the presentation of the manuscript.

References

[1] E. Andreeva, A. Bogdanov, A. Luykx, B. Mennink, E. Tischhauser, and K. Yasuda, "Parallelizable and authenticated online ciphers," in *Advances in Cryptology – ASIACRYPT 2013*, Part I, LNCS, vol. 8269, pp. 424–443, Springer, 2013.

[2] A. Bay, O. Ersoy, and F. Karakoç, "Universal forgery and key recovery attacks on ELmD authenticated encryption algorithm," in *Advances in Cryptology – ASIACRYPT 2016*, Part I, LNCS, vol. 10031, pp. 354–368, Springer, 2016. (Full version: IACR ePrint 2016/640.)

[3] E. Andreeva, A. Bogdanov, N. Datta, A. Luykx, B. Mennink, M. Nandi, E. Tischhauser, and K. Yasuda, "COLM v1," CAESAR Competition Round 3 Submission Specification, Sept. 15, 2016. Available at https://competitions.cr.yp.to/round3/colmv1.pdf.

[4] COLM Team, "Security of COLM," CAESAR Competition Round 3 Security Proof Addendum, 2016. Available at https://competitions.cr.yp.to/round3/colm-addendum.pdf.

[5] E. Andreeva, A. Bogdanov, N. Datta, A. Luykx, B. Mennink, M. Nandi, E. Tischhauser, and K. Yasuda, "The COLM authenticated encryption scheme," *Journal of Cryptology*, vol. 37, no. 2, art. 15, 2024.

[6] D. Chakraborty and M. Nandi, "COLM under attack: A cryptanalytic exploration of COLM variants," *Journal of Information Security and Applications*, vol. 89, art. 103936, 2025.

[7] N. Datta, A. Luykx, B. Mennink, and M. Nandi, "Understanding RUP integrity of COLM," *IACR Transactions on Symmetric Cryptology (ToSC)*, vol. 2017, no. 2, pp. 143–161, 2017. (Full version: IACR ePrint 2017/431.)

[8] N. Datta and M. Nandi, "ELmE: A misuse resistant parallel authenticated encryption," in *Information Security and Privacy – ACISP 2014*, LNCS, vol. 8544, pp. 306–321, Springer, 2014. (Full version: IACR ePrint 2013/767.)

[9] N. Datta and M. Nandi, "Proposal of ELmD v1.0," CAESAR Competition Round 1 Submission, 2014.

[10] C. Forler, E. List, S. Lucks, and J. Wenzel, "Reforgeability of authenticated encryption schemes," in *Information Security and Privacy – ACISP 2017*, Part II, LNCS, vol. 10343, pp. 19–37, Springer, 2017. (Full version: IACR ePrint 2017/332.)

[11] T. Krovetz and P. Rogaway, "The software performance of authenticated-encryption modes," in *Fast Software Encryption – FSE 2011*, LNCS, vol. 6733, pp. 306–327, Springer, 2011.

[12] J. Lu, "On the security of the COPA and Marble authenticated encryption algorithms against (almost) universal forgery attack," Cryptology ePrint Archive, Paper 2015/079, 2015.

[13] B. Mennink, "XPX: Generalized tweakable Even-Mansour with improved security guarantees," in *Advances in Cryptology – CRYPTO 2016*, Part I, LNCS, vol. 9814, pp. 64–94, Springer, 2016. (Full version: IACR ePrint 2015/476.)

[14] M. Nandi, "Revisiting security claims of XLS and COPA," Cryptology ePrint Archive, Paper 2015/444, 2015.

[15] S. E. Ulusoy, O. Kara, and M. Ö. Efe, "Plaintext recovery and tag guessing attacks on authenticated encryption algorithm COLM," *Journal of Information Security and Applications*, vol. 70, art. 103342, 2022.

[16] S. Vaudenay and D. Vizár, "Under pressure: Security of CAESAR candidates beyond their guarantees," Cryptology ePrint Archive, Paper 2017/1147, 2017.