More Keys, Less Commitment: Key-Committing Attacks on BBB Accordions via Encode-then-Encipher and a Key-Committing Transform

Donghoon Chang[1,2] Yukihito Hiraga[3],
Yusuke Naito[4], Yu Sasaki[2,5], and Takeshi Sugawara[3]

[1] FWI, Fairfax, USA, pointchang@gmail.com
[2] Associate of National Institute of Standards and Technology, Gaithersburg, USA
[3] The University of Electro-Communications, Tokyo, Japan,
yukihito.hiraga@uec.ac.jp, sugawara@uec.ac.jp
[4] Mitsubishi Electric Corporation, Kanagawa, Japan,
Naito.Yusuke@ce.MitsubishiElectric.co.jp
[5] NTT Social Informatics Laboratories, Tokyo, Japan, yusk.sasaki@ntt.com

Abstract. Cryptographic accordions with beyond-birthday-bound (BBB) security are being actively studied in connection with NIST's ongoing standardization effort. We show that BBB security of an accordion does not necessarily strengthen key commitment of AE built from it: in many BBB accordions, additional keys introduced for BBB security give an adversary extra degrees of freedom. Exploiting this observation, we present $O(1)$ key-committing attacks on a broad range of AE schemes built from BBB accordions, mostly via the Encode-then-Encipher (EtE) paradigm, including CHCTR2, eCTR-based schemes, THCTR, HCTR+, TCT_2, and bbb-ddd-AES under a natural polynomial-hash instantiation. To address this issue, we propose CACH, a black-box transform from BBB-secure accordions to robust AE (RAE) with key-committing security. CACH achieves BBB multi-user RAE security and achieves $t/2$-bit key-committing security with a t-bit tag for $n \leq t \leq 2n$ in the ideal-cipher model. CACH uses only AES-based components and supports all AES key sizes, making it compatible with NIST's design goals. We instantiate CACH with CHCTR2. CACH-CHCTR2 with AES-256 outperforms EtE-HCTR2 with Rijndael-256 and EtE-CHCTR2 for short messages.

Keywords: Key commitment · Accordion · Beyond-Birthday-Bound · HCTR2 · Encode-then-encipher · Polynomial hash · Blackbox transform

1 Introduction

Block ciphers (BCs) are typically used with a mode of operation for handling variable-length data. The earliest standardized modes, such as CBC and CTR [25], provide confidentiality only and are typically used with a separate message authentication code (MAC). Correctly composing these two primitives turned out to be nontrivial, and the composition is also costly, since it requires two independent keys and two full passes over the data. These security and efficiency

concerns motivated modes that deliver both confidentiality and integrity in a single, integrated design, namely authenticated encryption with associated data (AE) [54]. AE modes such as GCM [27], CCM [26], and OCB [44, 55] have since become the default interface of symmetric-key cryptography.

Cryptographic accordions, also known as variable-input-length strong tweakable pseudorandom permutation (VIL-STPRP), are drawing the attention of the research community as the National Institute of Standards and Technology (NIST) launched a standardization project in 2023, with its requirements documented in [11]. A long line of BC modes realizes accordions, including EME [33], HCTR [59], HCTR2 [16], and Adiantum [15]. Among the candidates, HCTR2 occupies a special position. It follows the hash-encrypt-hash structure instantiated with a polynomial hash (POLYVAL [32]) and an XCTR layer over AES; it is already deployed in the Linux kernel [57] and in Android file-based encryption [4]; and NIST has proposed developing accordion modes based on HCTR2 [52].

Many modes built on an n-bit BC inherit a common limitation: the security is lost once the number of processed blocks approaches $2^{n/2}$, because at that point collisions among internal n-bit values may occur and the PRP-PRF switching argument no longer holds. For AES, the birthday bound corresponds to only 64-bit security, which is insufficient for high-volume, long-lived, and multi-user deployments [42]. Overcoming the bound is one of the well-recognized challenges, and it has been pursued across various classes of modes.

NIST proposes to develop BBBAcc, BBB-secure accordion with AES, by extending HCTR2 [52]. Regarding the key size, NIST explicitly allows BBBAcc to support AES-128 and AES-192, in addition to the mandatory support for AES-256 [11]. Among existing constructions, CHCTR2, a cascade of two HCTR2 instances presented by Chen et al. [13], achieves $\frac{2n}{3}$-bit multi-user security, and is therefore particularly well aligned with the goals of BBBAcc. We consider it a primary candidate in this work. However, the study of BBB-secure accordions predates NIST's standardization effort, and several such constructions based on HCTR2 have been proposed independently of NIST. To distinguish the two notions, we use BBBAcc for the accordion to be standardized by NIST and BBB-accordion for any accordion with BBB security. We list several examples below. TCT_2 [56] builds an accordion with $\frac{2n}{3}$-bit security from AES and polynomial hashes. Some features such as fixed-length tweak and large keying material do not meet NIST's requirement, although these limitations appear amenable to modification. THCTR [24] replaces the BC in HCTR2 with a tweakable BC (TBC), aiming for single-user BBB security under a limited number of tweak repetitions, though the birthday attack was later presented by Andreeva et al. [2]. HCTR+ [19] also adopts an n-bit TBC and a TBC-based counter mode, together with a $2n$-bit polynomial hash, to achieve BBB security. bbb-ddd-AES [21] builds an accordion with $\frac{2n}{3}$-bit security under a limited number of tweak repetitions by using a deck function that is instantiated with the same primitive as those in HCTR2, i.e. AES and POLYVAL. eCTR-based accordion [14], which was initially presented at NIST 2023 workshop as DbHCTR [45], constructs a CENC-based BBB-secure PRF called eCTR, which is then extended to an accordion with $\frac{2n}{3}$-

bit security [14] by combining it with CTET^+, which we call CTET^+-eCTR, or five-round Feistel-network, which we call $\mathsf{F5}$-eCTR. These constructions provide a diverse set of BBB accordions for investigating whether the key-committing-security issue extends beyond a particular design.

According to NIST IR 8552 [11], an accordion is expected to support several *derived functions*, including AE. The efficient realization of robust AE (RAE) [3, 37] is a significant advantage of accordions, and it is obtained simply through the encode-then-encipher (EtE) framework [7]: the message is encoded with τ bits of redundancy, the encoded message is enciphered by the accordion with the nonce and associated data as the tweak, and decryption verifies the redundancy after deciphering. Zero padding is the most common encoding [37].

Committing security on AE is a property that restricts the decryption of a ciphertext to a certain set of parameters, such as a key, nonce, associated data, and plaintext. Particularly, key-committing security is relevant to real-world systems, including end-to-end encrypted messaging [22,29,39], key rotation in key-management services, envelope encryption [5, 35, 36], Google's Subscribe with Google [1], and password-based encryption and key exchange [46]. Key-committing security allows the adversary to know or even choose the key, which is a critical difference from the conventional AE notions. Farshim et al. [28] formalized the notion, which has since evolved through a series of works mainly for nonce-based AE [6, 47]. Meanwhile, EtE-accordion takes K, T, M and returns C for the encoded M. For this accordion syntax, Chang et al. [10] showed two kinds of key-committing security by adjusting the notions originally proposed for nonce-based AE by Menda et al. [47]; CMT_k that requires only $K = K'$ and $\mathsf{CMT}^*_\mathsf{k}$ that additionally requires $T = T'$. Importantly, NIST requires key-committing security for accordion-based AEs [11]. In this paper, we mainly target $\mathsf{CMT}^*_\mathsf{k}$ security in our attacks and CMT_k security in our proofs.

Subsequent work tackled committing security of AEs constructed from accordions. Chen et al. [12] initiated this direction by showing dedicated attacks on Adiantum [15], AEZ, and $\mathsf{HCTR2}$ with EtE. The position of the padding matters in committing security [10, 12]. The methods that append and prepend zeros to the message are referred to as EtE_A and EtE_P, respectively. The method called EtE_S [10] appends zeros to the second message block, similar to the one used for Adiantum. Their attack on EtE_P-$\mathsf{HCTR2}$ and EtE_A-$\mathsf{HCTR2}$ work with $O(2^{\min\{n/2,\tau\}})$ and $O(2^{\tau/2})$, respectively. Chang et al. [10] then reduced the complexity for EtE_P-$\mathsf{HCTR2}$ and EtE_A-$\mathsf{HCTR2}$ to $O(2^{\tau/2})$ and $O(2^{\max\{\tau/3,\,\tau-n\}})$, respectively. They showed that the $O(2^{\tau/2})$ attack on EtE_P-$\mathsf{HCTR2}$, as well as the $O(2^{\tau/2})$ attack on EtE_S-$\mathsf{HCTR2}$ is tight by proving matching bounds in the ideal-cipher model. As a result, the picture is now essentially complete with EtE_P-$\mathsf{HCTR2}$ and EtE_S-$\mathsf{HCTR2}$: for each encoding, the security is characterized as a function of τ and n, and it is settled by matching attacks and proofs.

All of the above attacks and proofs concern accordions with birthday-bound STPRP security, and the committing security of BBB accordions is a significant open problem. A natural expectation is that a BBB accordion, providing a higher STPRP security level, should provide improved committing security. This paper

Table 1. Attack complexity for AE constructed from HCTR2-based accordion. The reported complexity corresponds to the case in which the attack works most efficiently.

Security	Scheme	Best Attack	Ref
—	EtE-Generic	$O(2^\tau)$	—
BB	EtE-Adiantum	$O(2^{\tau/2})$	[12]
	EtE-HCTR2	$O(2^{\tau/3})$	[10]
BBB	EtE-CHCTR2	$O(1)$	Section 5.1
	EtE-CHCTR2-1k	$O(1)$	Section 5.2
	EtE-CHCTR2-1k$*$	$O(2^{\tau/3})$	Section 5.2
	EtE-CTET$^+$-eCTR	$O(1)$	Section 5.3
	EtE-F5-eCTR	$O(1)$	Section 5.4
	EtE-THCTR	$O(1)^\dagger$	Section 5.5
	EtE-HCTR+	$O(1)$	Section 5.6
	EtE-TCT$_2$	$O(1)$	Section 5.7
	aaa-bbb-ddd-AES	$O(1)^\ddagger$	Section 5.8

†The same attack applies to a ragged-PRP based AE scheme EtE-THEC in $O(1)$.
‡Assumed that a universal hash function family to process AD is a polynomial hash.
aaa is a dedicated transform for constructing AE from bbb-ddd-AES, inspired by EtE.

is motivated by the following questions. (i) When a BBB accordion such as CHCTR2 is turned into an AE via EtE, is its committing security better than, equal to, or worse than that of EtE-HCTR2, and how does the answer depend on the encoding? (ii) If it is worse, what is the commonality among insecure accordions regarding committing security? (iii) How can we improve committing security and salvage existing BBB-secure accordions without invasive changes?

1.1 Our Contributions

Generic key-committing attack approaches for EtE-*BBB-accordion*. We present $O(1)$ key-committing attacks on most of HCTR2-based BBB-secure accordions converted into AE via EtE (EtE-BBB-accordion). The attack exploits multiple independent keys, which do not cause problems in the conventional secret-key setting but provide additional degrees of freedom in the key-committing setting. Consequently, these schemes can be weaker than HCTR2 itself on key-committing attacks. We observe that this weakness commonly arises in approaches for achieving BBB security from HCTR2, including using a $2n$-bit IV, cascading HCTR2, and replacing the underlying primitive with a TBC: the first and third approaches often lead to independent hash keys due to different hash sizes, while cascading enables a new generic attack based on swapping the order of two HCTR2 instances.

Attacks on various EtE-*BBB-accordion schemes.* We evaluate the $\mathsf{CMT}^*_\mathsf{k}$-security of various EtE-BBB-accordion schemes. We find that most of them admit $O(1)$-

time attacks. This includes EtE with CHCTR2, CHCTR2-1k, CTET$^+$-eCTR, F5-eCTR, THCTR, HCTR+, and TCT$_2$. For bbb-ddd-AES, which proposes its own AE construction aaa-bbb-ddd-AES, we also give an $O(1)$-time attack. The attacks exploit additional independent keys introduced to achieve BBB security in the secret-key setting. These $O(1)$ attacks might be avoided by deriving all keys from a single key. CHCTR2-1k$*$, a natural extension of CHCTR2 with a single key, exhibits increasing security as the encoded length τ grows, although the best attack we found still has complexity below the birthday bound in τ. These results highlight the need for carefully designing AE to ensure CMT$_\mathsf{k}$ security in BBB-accordion.

CACH*: Blackbox transform from BBB-accordion to* CMT$_\mathsf{k}$*-secure AE.* Black-box transforms from accordions to committing-secure AE have been studied in previous work. EwC [38] constructs key-committing secure AE from an accordion, but it requires a BC with a $2n$-bit key as its underlying primitive, while NIST explicitly allows BBBAcc to support AES-128 and AES-192, in addition to AES-256. Constructions for achieving context-committing secure AE from an accordion, such as EwC2, HtE-EwC2 [38], and FFF [51], require cryptographically secure hash functions such as SHA-2, which goes beyond the goals of BBBAcc. We therefore propose CACH (CTR-Accordion-CENC-Hirose), a new transform for realizing RAE- and CMT$_\mathsf{k}$-secure AE in the multi-user setting, which supports all AES key sizes as the underlying BC.

If the underlying accordion provides s-bit security, CACH achieves $\min\{\frac{2n}{3}, s\}$-bit RAE security and $\frac{t}{2}$-bit CMT$_\mathsf{k}$ security for $n \le t \le 2n$ under the unrestricted online-query setting. In particular, if the underlying accordion provides n-bit security, under NIST's setting of at most $2^{n/2}$ queries per user, CACH with a $2n$-bit tag achieves n-bit security for both RAE and CMT$_\mathsf{k}$ notions. For comparison with EtE, which incurs τ bits of ciphertext expansion, we set $t = \tau$. Under the same ciphertext expansion, CACH achieves $\tau/2$-bit CMT$_\mathsf{k}$ security.

In a nutshell, CACH first derives keys, and then computes the accordion for a plaintext M, without applying encoding, to obtain a ciphertext C. Then, CACH applies a collision-resistant PRF (CR-PRF) to a $2n$-bit portion of $M \oplus C$, and its output is used as an AE tag. The CR property ensures CMT$_\mathsf{k}$-security by preventing tags generated under different keys from colliding. The PRF property ensures RAE-security: C and M are uniformly random in encryption and decryption, respectively; hence, for fresh queries, $M \oplus C$ is uniformly random, and so is the corresponding tag. Since our primary target is NIST's BBBAcc, we instantiate the CR-PRF using AES, particularly by designing a structure, which we call CEtH (CENC-then-Hirose).

We implement EtE-CHCTR2 and CACH-CHCTR2 instantiated with AES-256, optimized with the AES and CLMUL instructions, and benchmark them on an Intel Xeon w5-3423 (Sapphire Rapids) processor. For comparison, we also implement the birthday-bound scheme EtE-HCTR2 instantiated with the 256-bit BC Rijndael-256, as considered by NIST [52]. Our results show that both EtE-CHCTR2 and CACH-CHCTR2 outperform the Rijndael-256-based counterpart, since realizing Rijndael-256 with the AES instructions incurs additional SIMD overhead. Moreover, CACH-CHCTR2 is faster than EtE-CHCTR2 for short

messages despite its higher security level: the additional cost of CACH is smaller than the cost of processing the two extra message blocks introduced by the zero padding in EtE. For long messages, this constant difference becomes negligible and the two schemes converge to the same throughput.

2 Preliminaries

Notation. Let ε be the empty string and $\emptyset$ be the empty set. For an integer $i \geq 0$, let $\{0,1\}^i$ be the set of all i-bit strings and $\{0,1\}^0 \coloneqq \{\varepsilon\}$. Let $\{0,1\}^{\geq i}$ be the set of all bit strings of length $\geq i$. For a bit string X and an integer i, let $\mathsf{msb}_i(X)$ and $\mathsf{lsb}_i(X)$ be the most and least significant i bits of X, respectively. For integers $0 \leq i \leq j$, let $[i,j] \coloneqq \{i, i+1, \ldots, j\}$, and $[j] \coloneqq [1,j]$. If $j < i$ then $[i,j] \coloneqq \emptyset$. For a non-negative integer i, let $\mathtt{bin}_n(i)$ be the little-endian conversion of i to an n-bit string. For a value or a set X, $Y \leftarrow X$ means that X is assigned to Y. For a positive integer n and a bit string D of length $\geq n$, $(D_1, D_2) \xleftarrow{n,|D|-n} D$ means that D is partitioned into the n-bit string D_1 and the $(|D|-n)$-bit string D_2 such that $D = D_1 \| D_2$, and $(D_1, \ldots, D_\ell) \xleftarrow{n} D$ means that D is partitioned into the ℓ strings $D_1, \ldots, D_\ell$ such that $|D_i| = n$ for $i = 1, \ldots, \ell-1$, $|D_\ell| \leq n$, and $D = D_1 \| \cdots \| D_\ell$. For a non-empty set $\mathcal{T}$, $T \xleftarrow{\$} \mathcal{T}$ means that an element is chosen uniformly at random from $\mathcal{T}$ and assigned to T. For two sets $\mathcal{T}_1$ and $\mathcal{T}_2$, $\mathcal{T}_1 \xleftarrow{\cup} \mathcal{T}_2$ means that $\mathcal{T}_1 \leftarrow \mathcal{T}_1 \cup \mathcal{T}_2$. For a non-negative integer j and a set $\mathcal{S}$, let $\mathsf{UConf}_j(\mathcal{S}) \coloneqq \{\{S_1, S_2, \ldots, S_j\} \subseteq \mathcal{S}^j \mid \forall (i_1, i_2) \in [j]^2 : i_1 \neq i_2 \Rightarrow S_{i_1} \neq S_{i_2}\}$. For a set $\mathcal{S}$ of tuples of l elements and $j \in [l]$, let $(y_1, \ldots, y_{j-1}, *, y_{j+1}, \ldots, y_l) \in \mathcal{S}$ be a condition that $\exists y$ s.t. $(y_1, \ldots, y_{j-1}, y, y_{j+1}, \ldots, y_l) \in \mathcal{S}$.

Key-Committing Security. Key-committing security for the syntax of accordion-based AE via EtE was introduced by Chang et al. [10].

Definition 1. *Let* $\mathsf{EtE}[P]$ *be an* EtE *AE scheme based on an internal idealized primitive* P*. On input a key* K*, a tweak* T*, and a plaintext* M*,* $\mathsf{EtE}[P]$ *returns a ciphertext. The* $\mathsf{CMT}^*_{\mathsf{k}}$ *and* $\mathsf{CMT}_{\mathsf{k}}$ *advantages of* EtE *are defined as*

$$\mathbf{Adv}^{\mathsf{cmt}^*_{\mathsf{k}}}_{\mathsf{EtE}[P]}(\mathbf{A}) \coloneqq \Pr\left[((K,T,M),(K',T',M')) \leftarrow \mathbf{A}^P, \right.$$
$$\left. K \neq K', T = T', \mathsf{EtE}_K(T,M) = \mathsf{EtE}_{K'}(T',M')\right],$$
$$\mathbf{Adv}^{\mathsf{cmt}_{\mathsf{k}}}_{\mathsf{EtE}[P]}(\mathbf{A}) \coloneqq \Pr\left[((K,T,M),(K',T',M')) \leftarrow \mathbf{A}^P, \right.$$
$$\left. K \neq K', \mathsf{EtE}_K(T,M) = \mathsf{EtE}_{K'}(T',M')\right].$$

Assuming that a nonce and AD are encoded into a tweak, the $\mathsf{CMT}^*_{\mathsf{k}}$ and $\mathsf{CMT}_{\mathsf{k}}$ notions correspond to those defined by Menda et al. Since $\mathsf{CMT}^*_{\mathsf{k}}$ requires the tweak values in the input tuple to be identical, while $\mathsf{CMT}_{\mathsf{k}}$ does not, $\mathsf{CMT}_{\mathsf{k}}$ is a stronger notion. Note that the above notions naturally extend beyond EtE to any AE derivation from an accordion.

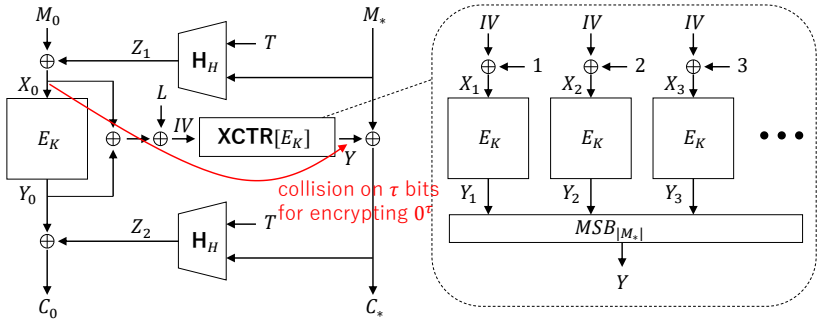

Fig. 1. HCTR2 and XCTR. $H = E_K(\mathtt{bin}_n(0))$ and $L = E_K(\mathtt{bin}_n(1))$. Red arrow is the core part of the key-committing attack.

3 HCTR2 Specification and Previous Attacks

3.1 Specification of HCTR2

HCTR2 [16] is an accordion constructed by a hash-encrypt-hash structure, where the hash is a family of keyed hash functions. $\mathsf{HCTR2}[E_K] : \{0,1\}^* \times \{0,1\}^{\geq n} \to \{0,1\}^{\geq n}$ denotes the HCTR2 encryption, which takes a tweak $T \in \{0,1\}^*$ and a plaintext $M \in \{0,1\}^{\geq n}$, and returns a ciphertext C such that $|M| = |C|$. HCTR2 uses the polynomial hash function H instantiated with a specific function POLYVAL, and the encryption part uses XCTR. H_H takes a key $H \in \{0,1\}^n$, a tweak $T \in \{0,1\}^*$, and a plaintext or ciphertext in $\{0,1\}^*$ and returns an n-bit output. The hash key is defined as $H = E_K(\mathtt{bin}_n(0))$. $\mathsf{XCTR}[E_K]$ is a mode of stream encryption that takes an n-bit IV and a data length and returns a key stream of the specified length. The encryption procedure of HCTR2 is shown in Alg. 1, which is also depicted in Fig. 1. In detail, a plaintext M is divided into an n-bit left part M_0 and the remaining right part M_*. In the first hashing layer, H_H is evaluated on T and M_* to produce the output Z_1. In the encryption part, the value $X_0 = M_0 \oplus Z_1$ is encrypted by the BC E_K to produce Y_0. The right part of the plaintext M_* is encrypted using XCTR with $IV = X_0 \oplus Y_0 \oplus L$, computing the right part of the ciphertext C_* by $Y \oplus M_*$, where Y is the XCTR's output. In the second hashing layer, H_H is evaluated on T and C_* to produce the output Z_2, and the left n-bit part of the ciphertext C_0 is obtained by XORing Z_2 with Y_0. The ciphertext is defined as $C = C_0 \| C_*$.

3.2 Inverting Polynomial Hash H

Suppose that a polynomial hash H pads the input as specified by HCTR2. Let poly_H be a function that returns $D_1 \cdot H^\ell \oplus D_2 \cdot H^{\ell-1} \oplus \cdots \oplus D_{\ell-1} \cdot H^2 \oplus D_\ell \cdot H$ for an ℓ-block input $D = D_1, \ldots, D_\ell$. Note that the $\overline{\mathsf{poly}}$ function for HCTR2 is a modified version of this function in order to optimize the performance by taking

Algorithm 1 Encryption of $\mathsf{HCTR2}$

Main $\mathsf{HCTR2}[E_K](T, M)$

1: $H \leftarrow E_K(\mathtt{bin}_n(0))$; $L \leftarrow E_K(\mathtt{bin}_n(1))$
2: $C \leftarrow \mathsf{HCTR2core}[E_K, \mathsf{H}](H, L, T, M)$
3: **return** C

Core Procedure $\mathsf{HCTR2core}[E_K, \mathsf{H}](H, L, T, M)$

1: $(M_0, M_*) \xleftarrow{n,|M|-n} M$
2: $Z_1 \leftarrow \mathsf{H}_H(T, M_*)$; $X_0 \leftarrow Z_1 \oplus M_0$; $Y_0 \leftarrow E_K(X_0)$
3: $IV \leftarrow X_0 \oplus Y_0 \oplus L$; $Y \leftarrow \mathsf{XCTR}[E_K](IV, |M_*|)$; $C_* \leftarrow Y \oplus M_*$
4: $Z_2 \leftarrow \mathsf{H}_H(T, C_*)$; $C_0 \leftarrow Z_2 \oplus Y_0$
5: $C \leftarrow C_0 \| C_*$; **return** C

Subroutine $\mathsf{XCTR}[E_K](IV, |D|)$

1: **for** $i = 1, \ldots, \lceil |D|/n \rceil$ **do** $Y_i \leftarrow E_K(IV \oplus \mathtt{bin}_n(i))$ **end for**
2: $Y \leftarrow \mathsf{msb}_{|D|}(Y_1 \| \cdots \| Y_{\lceil |D|/n \rceil})$; **return** Y

Subroutine $\mathsf{H}_H(T, D)$

1: **if** n **divides** $|D|$ **then** $Z \leftarrow \overline{\mathsf{poly}}_H(\mathtt{bin}_n(2|T|+2) \| \mathsf{zp}(T) \| D)$
else $Z \leftarrow \overline{\mathsf{poly}}_H(\mathtt{bin}_n(2|T|+3) \| \mathsf{zp}(T) \| \mathsf{ozp}(D))$ **end if**
2: **return** Z

Subroutine $\overline{\mathsf{poly}}_H(\tilde{D})$

1: $D_1, \ldots, D_\ell \xleftarrow{n} \tilde{D}$; $\overline{H} \leftarrow H \cdot \mathtt{x}^{-n}$
2: $Z \leftarrow D_1 \cdot \overline{H}^{\ell} \oplus D_2 \cdot \overline{H}^{\ell-1} \oplus \cdots \oplus D_{\ell-1} \cdot \overline{H}^{2} \oplus D_\ell \cdot \overline{H}$; **return** Z

into account the endian issue, but the security analysis for poly and $\overline{\mathsf{poly}}$ are essentially the same. Given a non-zero hash key H, input data D, and an output value Z, it is easy to find a tweak T such that $Z = \mathsf{H}_H(T, D)$ because of the algebraic structure of polynomial hash. The analysis when both $|T|$ and $|D|$ are multiples of n is as follows. The extension to the other case is straightforward.

$$\begin{aligned}
&\mathsf{H}_H(T, D) = Z \Leftrightarrow \mathsf{poly}_H(\mathtt{bin}_n(2|T|+2) \| T \| D) = Z, \\
&\quad \Leftrightarrow (2|T|+2) \cdot H^{\frac{|T|+|D|}{n}+1} \oplus \mathsf{poly}_H(T) \cdot H^{\frac{|D|}{n}} \oplus \mathsf{poly}_H(D) = Z, \\
&\quad \Leftrightarrow \mathsf{poly}_H(T) = \left(Z \oplus (2|T|+2) \cdot H^{\frac{|T|+|D|}{n}+1} \oplus \mathsf{poly}_H(D)\right) \cdot H^{-\frac{|D|}{n}},
\end{aligned}$$

where the right-hand side is reduced to a single value. One can set all but one block of T arbitrarily and compute the required value for the remaining block using simple algebra. Moreover, the analysis can be extended to find a common tweak $T = T'$ such that $\mathsf{H}_H(T, D) = Z$ and $\mathsf{H}_{H'}(T', D') = Z'$ are simultaneously satisfied for given values H, D, Z, H', D', and Z'. For this, the adversary needs to solve a system of two linear equations, and a solution can be found if at least two blocks of $T(= T')$ can be controlled. In general, i equations of the form $\mathsf{H}_{H_i}(T, D_i) = Z_i$ can be inverted if i blocks are controlled by the adversary.

3.3 Existing Attacks on EtE-HCTR2 and Variants

Several works have studied committing security of HCTR2-like structure. Chen et al. [12] initiated the study and showed that context committing security, where the adversary's goal is to find two distinct $(K, T, M), (K', T', M')$ colliding in the ciphertext, can be broken with $O(1)$ complexity by finding T, T' that collide in the polynomial hash under $K = K'$ and $M = M'$. Since then, the key-committing security, i.e. $\mathsf{CMT}^*_\mathsf{k}$ and CMT_k notions, have been studied. The following observation of Chen et al. [12], also used in subsequent works, is fundamental to these attacks and is illustrated by the red arrow in Fig. 1.

> *Suppose that the encoded τ bits are positioned in the right-hand-side of the message, e.g.* EtE_A *or* EtE_S*. Then, it is sufficient for an adversary to find left-hand-side BC input-key pairs $(X_0, K), (X'_0, K')$ that produce a collision on the key stream values for encrypting the encoded bits.*

With such $(X_0, K), (X'_0, K')$, τ bits of C_* corresponding to the encoded 0^τ also collide. Then, for any value of the other ciphertext bits in C_0 and C_*, the adversary can choose T, T', where $T = T'$, so that all the computations are consistent. The analysis is as follows.

Because $(X_0, K), (X'_0, K')$ are fixed, the corresponding Y_0, Y'_0, Y, Y' are fixed. For the second hash layer, the output of H is computed by $Z_2 \leftarrow Y_0 \oplus C_0$ and $Z'_2 \leftarrow Y'_0 \oplus C_0$. For any $C(= C_0 \| C_*)$, the adversary can jointly solve for a common tweak $T = T'$ using the two-equation inversion described in Sect. 3.2. For the first hash layer, the adversary first computes M_* by $M_* \leftarrow Y \oplus C_*$ and $M'_* \leftarrow Y' \oplus C_*$. Then, M_0 is simply computed by $M_0 \leftarrow \mathsf{H}_H(T, M_*) \oplus X_0$ and $M'_0 \leftarrow \mathsf{H}_{H'}(T', M'_*) \oplus X'_0$.

The cost of generating the τ-bit collision depends on the structure. For HCTR2, Chen et al. [12] generated the collision by the birthday attack, which requires a cost of $O(2^{\tau/2})$. Chen et al. also analyzed another hash-encrypt-hash scheme Adiantum [15], in which the stream cipher XChaCha12, an extension of ChaCha12 [8], is used instead of XCTR. Because the birthday attack works in general, the attack cost for EtE-Adiantum is also $O(2^{\tau/2})$. Chang et al. [10] reduced the cost of generating a τ-bit key-stream collision for EtE_A-HCTR2 to $O(2^{\tau/3})$ by exploiting the block-wise key-stream generation of XCTR. Recently, Chang et al. [9] have observed that if the IV mask L is omitted from HCTR2, the key-stream collision is generated at a cost of $O(1)$ for up to n bits.

If the encoded zero bits are positioned inside the first message block, a more careful analysis is required. For EtE_P-HCTR2, Chang et al. [10] showed that the birthday attack on the encoded 0^τ still applies, resulting in an $O(2^{\tau/2})$ attack.

In summary, finding $(X_0, K), (X'_0, K')$ that generate a collision on τ-bit key-stream values for encrypting encoded 0^τ can be a first step to attack key-committing security of EtE with an HCTR2-based accordion.

4 Generic Approaches for Attacking EtE-BBB-accordion

To develop generic attack approaches, we first generalize the design of HCTR2-based BBB-accordion. Here, BBB-security refers to the conventional secret-key security of the underlying accordion.

Categorization of design types. One of the reasons why HCTR2 has the birthday-bound security is that a collision on the n-bit IV occurs after $O(2^{n/2})$ queries. With this in mind, we identify the following three recurring design approaches of BBB-accordion in HCTR2-based designs.

1. The first type enlarges the size of IV to $2n$ bits. This also requires enlarging the size of the left-hand-side of the plaintext to $2n$ bits. For the encryption of the left-hand-side, $2n$-bit BBB-secure SPRP or STPRP is constructed from an n-bit BC. For the encryption of the right-hand-side, the XCTR mode is replaced with a BBB-secure key-stream generation. The size of the polynomial hash also needs to be enlarged to 2^n bits.
2. The second type maintains the size of IV to n bits, but in order to avoid the simple IV collision, the entire computation of the birthday-bound secure construction is iterated twice with different keys, e.g. by adopting the cascade construction, which ensures that RAE security is broken only when IV collisions occur simultaneously in two instances.
3. The third type changes the primitive from n-bit BC to n-bit TBC. The use of TBC helps to achieve BBB security for the plaintext-ciphertext conversion.

Possible attack approaches. We discuss possible attack approaches, i.e. attack templates, for each design type. The common theme is that additional independently controllable key material, introduced to overcome the birthday bound in the conventional secret-key setting, may provide additional degrees of freedom in the key-committing attack.

1. (a) The first type builds a BBB-secure $2n$-bit SPRP, denoted by DB, from an n-bit BC. This often requires introducing a new key K_{DB} that is independent of the other part. Then, the adversary can choose two distinct values only for K_{DB}, and use the same key outside DB. Suppose that IV for the key-stream generation is computed by XORing the input and output of DB. Then, if the adversary finds two keys $K_{\mathsf{DB}}, K'_{\mathsf{DB}}$ that map the same input to the same output in DB, a collision on IV is generated, which eventually generates a collision on the key stream. As explained in Sect. 3.3, this is sufficient to attack the scheme. Hence, the general attack approach is to find $X_0, K_{\mathsf{DB}}, K'_{\mathsf{DB}}$ such that $\mathsf{DB}_{K_{\mathsf{DB}}}(X_0) = \mathsf{DB}_{K'_{\mathsf{DB}}}(X_0)$.
   (b) The first type requires a polynomial hash of size $2n$ bits. Since the original hash key size of HCTR2 is n bits, the use of a $2n$-bit polynomial hash often motivates the designers to introduce an independent hash key. In this case, the possible attack approach is to find two different hash keys and inputs, e.g. T, such that their outputs collide. Then, by using the same keys and inputs in everywhere else, the ciphertexts collide.

2. For the second type, the encoded bits on the right-hand side are successively XORed with the key streams of the two BB-secure accordion instances. Since their order does not affect the resulting XOR, a natural attack approach is to swap the keys of the two instances. Note that if two keys for two instances are not independently chosen, e.g. both keys are generated from a single master key, such a simple attack approach would not work.
3. For the third type, the idea of "replacing a BC with a TBC" itself does not introduce a new structural weakness. Yet, it is worth investigating whether tweak processing introduces independently keyed components. In particular, when the output of the polynomial hash is used as the tweak of the underlying TBC, the hash size is determined by the tweak size of the TBC, which may have a different size from other polynomial hashes. Then designers may introduce independent keys, which can be a target of attacks.

5 Attacks on Various EtE-BBB-accordion based on HCTR2

5.1 $O(1)$ Attack on EtE-CHCTR2

CHCTR2 [13] is a cascaded construction of HCTR2, which computes two HCTR2 instances with two independent keys K_1, K_2. To be more specific, encryption of CHCTR2 for a tweak T and a message M generates a ciphertext C as $C \leftarrow \mathsf{HCTR2}[E_{K_1}](T, \mathsf{HCTR2}[E_{K_2}](T, M))$, which is depicted in Fig. 2, left.

Let $(X_{1,0}, Y_{1,0})$ and $(X_{2,0}, Y_{2,0})$ be respectively the pairs of plaintext and ciphertext of E_{K_1} and E_{K_2} in the left part. Suppose that M_* consists of r blocks. For $j \in [r]$, let $(X_{1,j}, Y_{1,j})$ and $(X_{2,j}, Y_{2,j})$ be, respectively, the j-th pairs of E_{K_1} and E_{K_2} in the XCTR invocation, and Y_1 and Y_2 are the entire key stream in each instance. Let $Z_1, Z_{1,2}$, and Z_2 be respectively the hash values of the first, second, and third hash functions. $Z_{1,2}$ is the result of the sum of two hash values.

Attack Strategy. Suppose that the EtE_S is adopted in CHCTR2, namely the plaintext has 0^τ from the second block, or $\mathsf{msb}_\tau(M_*) = 0^\tau$. Then, 0^τ is encrypted as $\mathsf{msb}_\tau(C_*) = \mathsf{msb}_\tau(Y_1) \oplus \mathsf{msb}_\tau(Y_2) \oplus 0^\tau$. The adversary searches for another paired keys (K_1', K_2'), and the ciphertext for the encoded bits is $\mathsf{msb}_\tau(C_*') = \mathsf{msb}_\tau(Y_1') \oplus \mathsf{msb}_\tau(Y_2') \oplus 0^\tau$. Two ciphertexts needs to collide, thus the pair needs to have the key stream Y_1, Y_2, Y_1', Y_2' that satisfy the following 4-sum problem;

$$\mathsf{msb}_\tau(Y_1) \oplus \mathsf{msb}_\tau(Y_2) \oplus \mathsf{msb}_\tau(Y_1') \oplus \mathsf{msb}_\tau(Y_2') = 0. \tag{1}$$

$\mathsf{msb}_\tau(Y_1)$ is computed by fixing K_1 and the left-hand-side BC's input/output $X_{1,0}, Y_{1,0}$. Similarly for $\mathsf{msb}_\tau(Y_2)$, $\mathsf{msb}_\tau(Y_1')$, and $\mathsf{msb}_\tau(Y_2')$, $(K_2, X_{2,0}, Y_{2,0})$, $(K_1', X_{1,0}', Y_{1,0}')$, and $(K_2', X_{2,0}', Y_{2,0}')$ are fixed together, respectively. Therefore, when Y_1, Y_2, Y_1', Y_2' satisfying Eq. (1) is found, the keys and $(X_{1,0}, Y_{1,0})$, $(X_{2,0}, Y_{2,0})$, $(X_{1,0}', Y_{1,0}')$, and $(X_{2,0}', Y_{2,0}')$ are all fixed. Hence, the adversary needs to choose T and T' so that those values are consistent. If $T = T'$ is satisfied, the $\mathsf{CMT}_\mathsf{k}^*$ notion is attacked. If $T \neq T'$ is needed, the CMT_k notion is attacked. Such T and T' can be found by solving simple linear algebra. Suppose that $\mathsf{msb}_\tau(C_*)$, which is

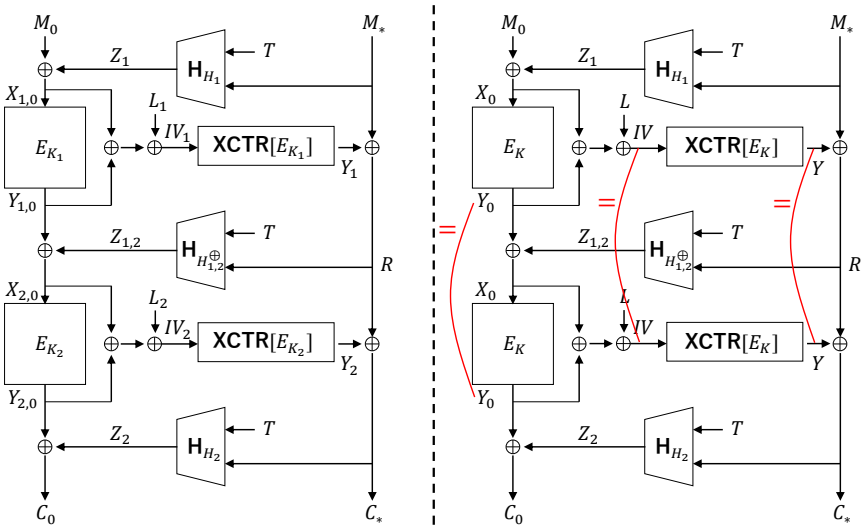

Fig. 2. Left: CHCTR2 specification, Right: CHCTR2-1k specification and its attack configuration. $\mathsf{H}_{H^{\oplus}_{1,2}} = \mathsf{H}_{H_1} \oplus \mathsf{H}_{H_2}$.

$\mathsf{msb}_\tau(Y_1) \oplus \mathsf{msb}_\tau(Y_2)$ or equivalently $\mathsf{msb}_\tau(Y_1)' \oplus \mathsf{msb}_\tau(Y_2)'$ is fixed. Then, the adversary fixes the other bits of $C(=C')$ arbitrarily. This fixes the right-hand-side of the state between two HCTR2 instances, R and R'. Moreover, it fixes $Z_2 = C_0 \oplus Y_{2,0}$, $Z'_2 = C_0 \oplus Y'_{2,0}$, $Z_{1,2} = X_{2,0} \oplus Y_{1,0}$, $Z'_{1,2} = X'_{2,0} \oplus Y'_{1,0}$. Eventually, it gives a system of 4 linear equations; $Z_2 = \mathsf{H}_{H_2}(T, C_*)$, $Z'_2 = \mathsf{H}_{H'_2}(T', C_*)$, $Z_{1,2} = \mathsf{H}_{H_1 \oplus H_2}(T, R)$, $Z'_{1,2} = \mathsf{H}_{H'_1 \oplus H'_2}(T', R')$. As mentioned in Sect. 3.2, with independently chosen H_1, H_2, H'_1, H'_2, a solution with $T = T'$ can be found by controlling 4 tweak blocks. When (H_1, H_2) and (H'_1, H'_2) are dependent, a solution with $T = T'$ may not be found. Even in this case, a solution with $T \neq T'$ can be found easily. T that satisfies both $Z_2 = \mathsf{H}_{H_2}(T, C_*)$ and $Z_{1,2} = \mathsf{H}_{H_1 \oplus H_2}(T, R)$ can be found by controlling 2 blocks of T, and the same applies to T' that satisfies both $Z'_2 = \mathsf{H}_{H'_2}(T', C_*)$ and $Z'_{1,2} = \mathsf{H}_{H'_1 \oplus H'_2}(T', R')$.

In summary, for any value of BC's input/output in the left-hand-side, the adversary can easily compute T and T' so that all the related values are consistent. The hardest part of the attack is to find $(K_1, X_{1,0}, Y_{1,0})$, $(K_2, X_{2,0}, Y_{2,0})$, $(K'_1, X'_{1,0}, Y'_{1,0})$, and $(K'_2, X'_{2,0}, Y'_{2,0})$ such that $\mathsf{msb}_\tau(Y_1) \oplus \mathsf{msb}_\tau(Y_2) \oplus \mathsf{msb}_\tau(Y'_1) \oplus \mathsf{msb}_\tau(Y'_2) = 0$.

Hereafter, we discuss this cost for CHCTR2, CHCTR2-1k, and CHCTR2-1k$*$.

Satisfying 4-sum in $O(1)$ for CHCTR2. By following the general approach in Sect. 4, we swap the order of the keys for two HCTR2 instances.

Algorithm 2 Attack Procedure for CHCTR2 with $\mathsf{EtE_A}$ or $\mathsf{EtE_S}$ Encoding

Set M_* to any value satisfying $|M_*| \geq \tau$ and including 0^τ properly; $C_0 \leftarrow \{0,1\}^n$

Set up for the first text

$K_1, K_2 \leftarrow \mathcal{K}$ s.t. $K_1 \neq K_2$; Derive H_1, H_2, L_1, and L_2 accordingly.

$X_{1,0} \xleftarrow{\$} \{0,1\}^n$; $Y_{1,0} \leftarrow E_{K_1}(X_{1,0})$; $Y_1 \leftarrow \mathsf{XCTR}[E_{K_1}](X_{1,0} \oplus Y_{1,0} \oplus L_1, |M_*|)$

$X_{2,0} \xleftarrow{\$} \{0,1\}^n$; $Y_{2,0} \leftarrow E_{K_2}(X_{2,0})$; $Y_2 \leftarrow \mathsf{XCTR}[E_{K_2}](X_{2,0} \oplus Y_{2,0} \oplus L_2, |M_*|)$

$R = M_* \oplus Y_1$; $C_* = R \oplus Y_2$; $Z_{1,2} = Y_{1,0} \oplus X_{2,0}$; $Z_2 = Y_{2,0} \oplus C_0$

Set up for the second text

$(K'_1, H'_1, L'_1) \leftarrow (K_2, H_2, L_2)$; $(K'_2, H'_2, L'_2) \leftarrow (K_1, H_1, L_1)$

$X'_{1,0} \leftarrow X_{2,0}$; $Y'_{1,0} \leftarrow Y_{2,0}$; $Y'_1 \leftarrow Y_2$

$X'_{2,0} \leftarrow X_{1,0}$; $Y'_{2,0} \leftarrow Y_{1,0}$; $Y'_2 \leftarrow Y_1$

$R' = C_* \oplus Y'_2$; $M'_* = M_*$; $Z'_{1,2} = Y'_{1,0} \oplus X'_{2,0}$; $Z'_2 = Y'_{2,0} \oplus C_0$

Find T, T' by inverting polynomial hash H

Find T satisfying both $Z_{1,2} = \mathsf{H}_{H_1}(T, R) \oplus \mathsf{H}_{H_2}(T, R)$ and $Z_2 = \mathsf{H}_{H_2}(T, C_*)$.

Find T' satisfying both $Z'_{1,2} = \mathsf{H}_{H_1}(T', R') \oplus \mathsf{H}_{H_2}(T', R')$ and $Z'_2 = \mathsf{H}_{H_1}(T', C_*)$.

$M_0 \leftarrow \mathsf{H}_{H_1}(T, M_*) \oplus X_{1,0}$

$M'_0 \leftarrow \mathsf{H}_{H_2}(T', M'_*) \oplus X'_{1,0}$

return $((K_1, K_2), T, (M_0 \| M_*)), ((K_2, K_1), T', (M'_0 \| M'_*))$

The adversary first chooses any keys K_1 and K_2 such that $K_1 \neq K_2$ and any BC's input $X_{1,0}$ and $X_{2,0}$, then computes Y_1 and Y_2 under K_1, K_2. Note that we have $\mathsf{msb}_\tau(C_*) = \mathsf{msb}_\tau(Y_1) \oplus \mathsf{msb}_\tau(Y_2)$.

For the second text, the adversary sets $(K'_1, K'_2) \leftarrow (K_2, K_1)$. This gives $(L'_1, L'_2) \leftarrow (L_2, L_1)$. The adversary also sets $X'_{1,0} \leftarrow X_{2,0}$ and $X'_{2,0} \leftarrow X_{1,0}$. This ensures that $Y'_1 = Y_2$ and $Y'_2 = Y_1$. Then, $\mathsf{msb}_\tau(C'_*)$ is computed by $\mathsf{msb}_\tau(C'_*) = \mathsf{msb}_\tau(Y'_1) \oplus \mathsf{msb}_\tau(Y'_2) = \mathsf{msb}_\tau(Y_2) \oplus \mathsf{msb}_\tau(Y_1)$, which collides with $\mathsf{msb}_\tau(C_*)$. The attack cost is $O(1)$.

Note that due to the dependence of hash keys, $H'_1 = H_2$ and $H'_2 = H_1$, it is unlikely that the solution can be found with $T = T'$. Hence, the adversary's goal for this attack is to find T, T' with $T \neq T'$, i.e. to break $\mathsf{CMT_k}$ security.

The attack procedure is described in Alg. 2 in detail. The attack was implemented and the generated concrete attack vectors are given in Table 2.

5.2 $O(1)$ Attack on EtE-CHCTR2-1k and $O(2^{n/3})$ Attack on EtE-CHCTR2-1k∗

Satisfying 4-sum in $O(1)$ for CHCTR2-1k. CHCTR2-1k is a variant of CHCTR2 that was presented by Naito in the public comment to NIST [50], where two HCTR2 instances use the same BC key K and hash keys are derived from K. Besides, based on the observation by Chen et al. [13], where the use of L is unnecessary for STPRP security of HCTR2, CHCTR2-1k removed L from two HCTR2 instances. The pseudo-code of the algorithm is given in Alg. 3 and the construction is shown in Fig. 2, right.

Table 2. Concrete vector for EtE-CHCTR2 with EtE_S encoding and $\tau = 96$.

Parameter	Value
K_1	AF501AB26EA6A65F3CF4E9C42C2C9191
K_2	82DC13460B1B33C35E2F7AB8C8BEFB40
T	1984412C4ED1F5A0201D1B0CCFE384D4 8F1F1CE4FC12D20FCD06146CF1078A80
M	97675573EA66FEDEA10F1B71A1D47D29 00000000000000000000000042925DB1 88DFEDEF80F95F028ED95BACBBE553DC
C	479D86F7A92615BC3821CEA83534BA3B C3D06AABD2598933D2693FAA33A3F72F C880E270527BBA23B76116044FF8B00F
K_1'	82DC13460B1B33C35E2F7AB8C8BEFB40
K_2'	AF501AB26EA6A65F3CF4E9C42C2C9191
T'	EDCC9246D2177B5A78B324AAB71A72FB 2DA5BAAC1770696E9E01CABEC5E2A7CA
M'	97675573EA66FEDEA10F1B71A1D47D29 00000000000000000000000042925DB1 88DFEDEF80F95F028ED95BACBBE553DC
C'	479D86F7A92615BC3821CEA83534BA3B C3D06AABD2598933D2693FAA33A3F72F C880E270527BBA23B76116044FF8B00F

Algorithm 3 Encryption of CHCTR2-1k

Encryption CHCTR2-1k$[E_K, \mathsf{H}](T, M)$

1: $H_1 \leftarrow E_K(\mathtt{bin}_n(0))$; $H_2 \leftarrow E_K(\mathtt{bin}_n(2))$;
2: $D \leftarrow \mathsf{HCTR2core}[E_K, \mathsf{H}](H_1, 0^n, T, M)$
3: $C \leftarrow \mathsf{HCTR2core}[E_K, \mathsf{H}](H_2, 0^n, T, D)$
4: **return** C

CHCTR2-1k uses the same key for BC in two HCTR2 instances, hence exchanging two keys adopted for attacking EtE-CHCTR2 does no longer work. However, the use of the same BC key enables another $O(1)$-cost attack. For this, the values of X_0 in two instances are set to be identical. This derives the same IV and moreover, the same key stream in two instances. The property holds for any K and X_0. Eq. (1) is satisfied in an $O(1)$ cost.

$O(2^{\tau/3})$ Attack on EtE-*CHCTR2-1k∗*. We consider CHCTR2-1k∗, a variant of CHCTR2-1k, where keys for two HCTR2 instances K_1, K_2 are generated by a KDF from a master key K, and two HCTR2 instances are not modified from the original, namely L is used to generate IV. For this construction, the best approach so far is to solve Eq. (1) by using the generalized birthday attack [58]. In fact, for fixed K, K' that also fix K_1, K_2, K_1', K_2', the functions to compute each of 4 terms are independent. The generalized birthday attack finds a solution of the 4-sum problem in $O(2^{\tau/3})$ time and memory.

5.3 $O(1)$ Attack on EtE-CTET$^+$-eCTR

Chung et al. presented a BBB-secure PRF called eCTR and its extension to BBB-secure accordion [14]. The main differences from HCTR2 are as follows.

– The size of the left-hand-side, i.e. $|M_0|$, is $2n$ bits.

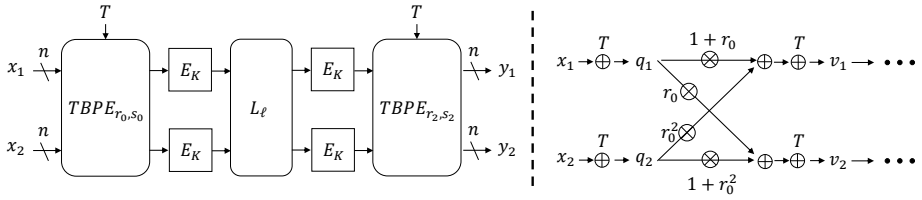

Fig. 3. CTET^+ (left) and beginning of first TBPE (right).

– The BC in the left-hand-side is replaced with DB; a $2n$-bit BBB-secure SPRP constructed from an n-bit BC.
– The IV size is $2n$ bits and XCTR is replaced with eCTR.

Chung et al. showed two instances for DB. This section analyzes one of two instances, where the DB is instantiated with CTET^+, which is a BC-based fixed-length-input tweakable SPRP having $\frac{2n}{3}$-bit security for $2n$-bit input. Hereafter, the construction is denoted by CTET^+-eCTR.

CTET^+ for $2n$-bit inputs requires $5n$-bit key denoted by r_0, s_0, ℓ, r_2, s_2. The input $x_1\|x_2$ is first linearly mixed by TBPE that takes as input a tweak T and $2n$-bit key (r_0, s_0). We omit the rest of CTET^+. The full CTET^+ is depicted in left-hand-side of Fig. 3. The right-hand-side of Fig. 3 depicts the first part TBPE with (r_0, s_0), where the part involving s_0 is not included.

We adopt the generic approach discussed in Sect. 4 for $2n$-bit IV, namely we aim to find $x_1\|x_2$ and a pair of keys that are used only inside CTET^+, namely $(r_0, s_0, \ell, r_2, s_2)$ that collide in CTET^+'s output. Moreover, we found that choosing two distinct r_0, r_0' is sufficient to generate a collision at the beginning of TBPE, so after an internal state collision occurs, by using the same key for (s_0, ℓ, r_2, s_2) and the same tweak T, the same $y_1\|y_2$ can be generated.

Let q_1 and q_2 be $x_1 \oplus T$ and $x_2 \oplus T$, respectively. Also, let z be $r_0 q_1 + r_0^2 q_2$. Then, we have $v_1 = x_1 + z$ and $v_2 = x_2 + z$. The adversary's goal is to find r_0, r_0' such that $v_1 = v_1'$ and $v_2 = v_2'$ for a commonly fixed $x_1\|x_2$, which is equivalent to find $z = z'$. This can be obtained easily by simple algebra.

From $z = z'$, we have $r_0 q_1 + r_0^2 q_2 = r_0' q_1 + r_0'^2 q_2$. Due to the property of the binary filed, we have $r_0^2 + r_0'^2 = (r_0 + r_0')^2$. Hence $q_1(r_0 + r_0') + q_2(r_0 + r_0')^2 = 0$, which is converted into $(r_0 + r_0')(q_1 + q_2(r_0 + r_0')) = 0$. The former is 0 results in the trivial solution $r_0 = r_0'$. The latter is 0 shows the non-trivial solution of $r_0' = r_0 + \frac{q_1}{q_2}$ when $q_1, q_2 \neq 0$. Hence, choose $(x_1\|x_2, T)$ such that $q_1, q_2 \neq 0$. Then r_0, r_0' that result in an internal collision is computed with an $O(1)$ cost.

5.4 $O(1)$ Attack on EtE-F5-eCTR

This section analyzes another eCTR-based Accordion instance [14], where the DB is instantiated with five-round Feistel network, denoted by $\mathsf{F5}$. $\mathsf{F5}$ is a $2n$-bit

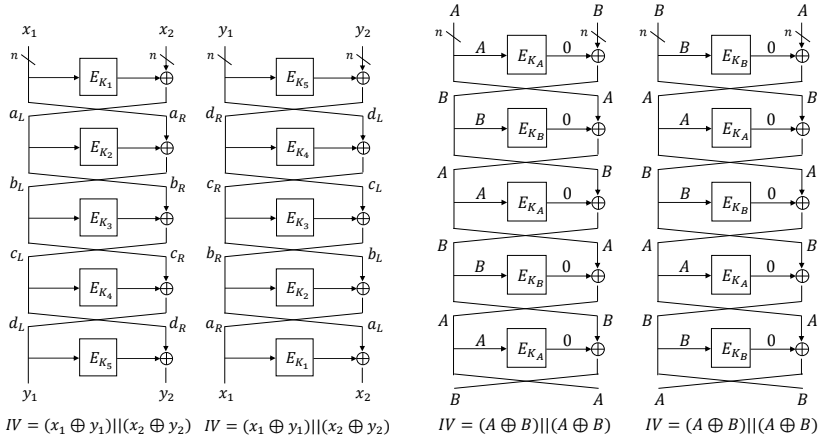

Fig. 4. Attacks without Last Network Swap in F5-eCTR

Fig. 5. Attacks with Last Network Swap in F5-eCTR

SPRP, where n-bit BC with independent key $K_i, i \in \{1, 2, 3, 4, 5\}$ is computed in each Feistel round. The construction is denoted by F5-eCTR.

We again adopt the generic approach discussed in Sect. 4, in which we find a pair of keys that are used only inside F5, namely $(K_1, K_2, K_3, K_4, K_5)$. Differently from CTET$^+$-eCTR, here we generate the IV collision with different input and output values. The IV for eCTR is computed by $x \oplus \mathsf{F5}_K(x)$. Hence, our goal is to find (x, K) and (x', K') such that $x \oplus \mathsf{F5}_K(x) = x' \oplus \mathsf{F5}_{K'}(x')$. Note that we need additional effort to prevent the modified F5's output from impacting C_0. This can be done by modifying T as explained later.

The exact attack procedure is significantly affected by a subtle difference of the definition; whether or not the five-round Feistel network adopts the network swap in the last round. Chung et al. [14] did not specify such details, and we would consider attacks in both settings.

Analysis without the swap. When F5 does not have the last network swap, an $O(1)$ attack is possible by utilizing the involution property of Feistel network. The attack is depicted in Fig. 4. The adversary first chooses any key $(K_1, K_2, K_3, K_4, K_5)$ satisfying $K_1\|K_2 \neq K_5\|K_4$. Then, for any $2n$-bit input $x_1\|x_2$, the output $y_1\|y_2$ under $(K_1, K_2, K_3, K_4, K_5)$ is computed. The IV is the XOR of the input and output, hence $IV = (x_1 \oplus y_1)\|(x_2 \oplus y_2)$. For the second text, the adversary chooses the key in the reverse order $(K_5, K_4, K_3, K_2, K_1)$, and exchanges the input and the output, i.e. the input is $y_1\|y_2$. Due to the involution property, the five-round computation of the second pair behaves as the decryption of the first pair, which returns $x_1\|x_2$. Then, the adversary obtains $IV = (x_1 \oplus y_1)\|(x_2 \oplus y_2)$, which collides with the IV for the first text.

Y_0 changed from $y_1\|y_2$ to $x_1\|x_2$, i.e. the difference $x_1 \oplus y_1 \| x_2 \oplus y_2$ exists in two texts. Since C_* has no difference, we need to cancel this difference by adjusting T in the second hash layer. This can be done with $O(1)$ cost by inverting the polynomial hash function. The change of T in the third layer gives the same change in the first layer. Since we generated $X_0 \oplus Y_0 = X'_0 \oplus Y'_0$, we have $X_0 \oplus X'_0 = Y_0 \oplus Y'_0$. In fact, X_0 also has the difference $x_1 \oplus y_1 \| x_2 \oplus y_2$. Hence, interestingly, if $C_0 = C'_0$ is ensured by modifying T, $M_0 = M'_0$ is also ensured.

Analysis with the swap. When F5 adopts the last network swap, the same attack cannot work. This is because the first text becomes $(x_1\|x_2) \to (y_2\|y_1)$ and the second text becomes $(y_1\|y_2) \to (x_2\|x_1)$. The corresponding IVs are $(x_1 \oplus y_2)\|(x_2 \oplus y_1)$ and $(x_2 \oplus y_1)\|(x_1 \oplus y_2)$. Those collide only when the left and right halves of IV are equal, which requires a $O(2^{n/2})$ cost. We introduce another idea depicted in Fig. 5. For any two distinct keys K_A and K_B, the 5-round key is set to $(K_A, K_B, K_A, K_B, K_A)$. Then, two n-bit values A, B are computed as $A \leftarrow E^{-1}_{K_A}(\mathtt{bin}(0))$ and $B \leftarrow E^{-1}_{K_B}(\mathtt{bin}(0))$, and the input value to F5 is set to $A\|B$. This configuration achieves a 2-round iterative pattern, where the state $A\|B$ becomes $B\|A$ after 1 round, and gets back to $A\|B$ after 2 rounds. Hence, the output after 5 rounds is $B\|A$, and $IV = (A \oplus B)\|(A \oplus B)$. For the second pair, we set the 5-round key to $(K_B, K_A, K_B, K_A, K_B)$ and set the plaintext to $B\|A$. Again, this achieves a 2-round iterative pattern. The output after 5 rounds is $A\|B$ and $IV = (A \oplus B)\|(A \oplus B)$, which collides with the IV of the first text.

To ensure no difference exists C_*, we modify T to cancel the difference in Y_0, similarly to the case with the last network swap.

Remarks. The attack on F5 without the last swap exploits the involution property, and the attack on F5 with the last swap exploits the iterative pattern. Both property holds for any number of rounds inside the Feistel network. Hence, both attacks can be extended to any number of Feistel rounds.

5.5 $O(1)$ Attack on EtE-THCTR

THCTR (tweakable HCTR2) [24] is a BBB-accordion that uses a TBC $\tilde{E_K}$ instead of a BC E_K as a primitive, and use the TBC-based counter mode $Ctr[\tilde{E_K}]$ instead of XCTR, which provides BBB security under a limited number of repetitions of the same tweak. The construction is depicted in Fig. 6.

The important modification from HCTR2 is that the tweak T is not processed inside the n-bit polynomial hash H. It is rather processed by introducing another $n+m$-bit polynomial hash H′, where m is the tweak size of the underlying TBC. Also, an independent $n+m$-bit hash key L is introduced for processing T.

Following the approach for TBC-based schemes shown in Sect. 4, we analyze this new tweak process. Since T is processed by an independent hash key L, our attack uses distinct values only for L and L'. The TBC key K and n-bit hash key H stay unchanged between two texts. Then, the adversary finds T such that $\mathsf{H}'_L(T) = \mathsf{H}'_{L'}(T)$ with an $O(1)$ cost by solving two linear equations. Then for any K, H, M, the ciphertext for $\mathsf{THCTR}(T, M)$ with L and L' result in the same ciphertext, which breaks the $\mathsf{CMT}^*_{\mathsf{k}}$ security.

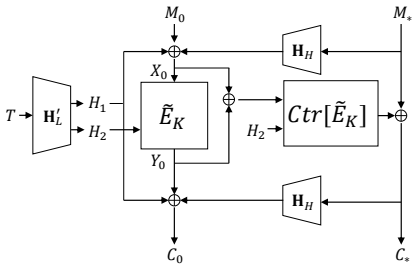

Fig. 6. THCTR. L is an independent key.

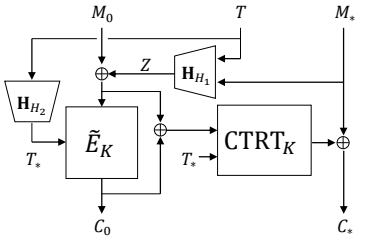

Fig. 7. THEC. H_1 and H_2 are independent keys.

Extension to $O(1)$ Attack on EtE-THEC. Rugged PRP (RPRP) is a security notion for VIL-TBC proposed by Degabriele and Karadžić at CRYPTO 2022 [20], which is strictly weaker than the traditional notion of a STPRP. They showed that by applying EtE to RPRP, an AE mode that satisfies the notion of misuse-resistant AE (MRAE) and releasing-unverified-plaintext-secure AE (RUP-secure AE) is achieved. Subsequent work by Datta et al. [18] has further studied RPRPs, including constructions achieving BBB security under the assumption that the number of the same tweak repetitions is limited.

Datta et al. [18] showed a specific instance called THEC to satisfy the RPRP notion, which is depicted in Fig. 7. A notable difference is that the last hash layer is omitted from the hash-encrypt-hash paradigm. Other than that, the structure has a certain similarity as that of THCTR. In particular, to process T, two hash keys H_1, H_2 that are independent from the TBC key are used.

The attack approach for EtE-THEC is similar to one for EtE-THCTR. The adversary uses different values only for two hash keys H_1, H_2. The exact attack step is as follows. The adversary first chooses any key K, a message M, and two hash outputs T_*, Z to compute the corresponding C. The adversary then chooses two hash keys $(H_1, H_2), (H'_1, H'_2)$ and finds a T that produces T_*, Z under both (H_1, H_2) and (H'_1, H'_2). This is done by inverting a polynomial hash function, and it is sufficient to control four blocks of T. The attack cost is $O(1)$.

5.6 $O(1)$ Attack on EtE-HCTR+

HCTR+ [19] adopts an n-bit block TBC as an underlying primitive, and adopts a TBC-based counter mode $\widetilde{CTRT}$ to generate the key stream as shown in Fig. 8. Two hash layers use a polynomial hash of size $2n$ bits and a $2n$-bit hash key H that is independent from the other TBC keys is introduced.

Our attack approach is only using distinct values H, H' for hash key. The TBC keys stay unchanged between two texts in the pair. For some M, the adversary finds a tweak T that produces the same results both in $\mathsf{H}_H(T, M)$ and $\mathsf{H}_{H'}(T, M)$. Then for any $K_1, \ldots, K_4$, the ciphertext for HCTR+(T, M) with $(K_1, \ldots, K_4, H)$ and $(K_1, \ldots, K_4, H')$ generate the same ciphertext.

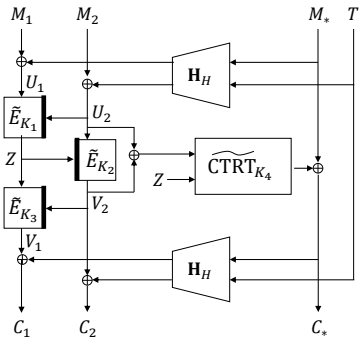

Fig. 8. HCTR+. A hash key H is independent from the other keys.

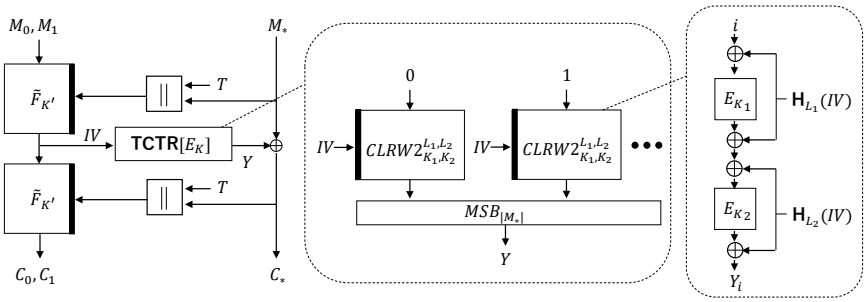

Fig. 9. TCT_2. Hash keys L_1, L_2 inside CLRW2 are independent from other part.

5.7 $O(1)$ Attack on EtE-TCT_2

TCT_2 [56] is a BBB-accordion designed by Shrimpton and Terashima at Asiacrypt 2013 as a PIV construction. The construction of TCT_2 is depicted in Fig. 9.

Since TCT_2 was designed early, there are several differences from HCTR2. The important part for key-committing security is that the key-stream to encrypt M_* is generated with the CLRW2 construction that takes $2n$-bit IV as a tweak and the counter value as a plaintext. Inside, CLRW2, the value of IV is processed by a polynomial hash with two independent keys L_1 and L_2. More importantly, those keys are independent from the other part of TCT_2.

Our attack approach is only using distinct values for L_1. The adversary first randomly chooses a key, a message, and a tweak, and then compute TCT_2 to obtain the ciphertext. Let $IV_1 \| IV_2 = IV$. The attack needs $IV_1 \neq 0$ and $IV_2 \neq 0$ for the following analysis, hence if IV_1 or IV_2 is 0, the input values are chosen again. To generate the second text, the same message and tweak are reused, and for the key, only L'_1 is different from L_1. The adversary's goal is to compute L'_1 so that $\mathsf{H}_{L_1}(IV) = \mathsf{H}_{L'_1}(IV)$. This can be done with a cost of $O(1)$ as follows.

We find L'_1 s.t. $IV_1 \cdot L_1^2 \oplus IV_2 \cdot L_1 \oplus IV_1 \cdot (L'_1)^2 \oplus IV_2 \cdot L'_1 = 0$. Then we have $IV_1(L_1 \oplus L'_1)^2 \oplus IV_2(L_1 \oplus L'_1) = 0$, which gives $(L_1 \oplus L'_1)(IV_1(L_1 \oplus L'_1) \oplus IV_2) = 0$. The non-trivial case $IV_1(L_1 \oplus L'_1) \oplus IV_2 = 0$ leads to $L'_1 = \frac{IV_2}{IV_1} \oplus L_1$. Hence, L'_1 can be computed with a cost of $O(1)$.

5.8 $O(1)$ Attack on aaa-bbb-ddd-AES

bbb-ddd-AES [56] is a BBB-accordion that takes different approach from NIST's standardization plan. The design is based on the deck function and follows a Hash–PRF–PRF–Hash structure, rather than the hash–encrypt–hash paradigm. It instantiates the hash with POLYVAL and the PRF with an AES-based XE-style construction, achieving the competitive performance with HCTR2.

bbb-ddd-AES supports tweaks of up to 96 bits and therefore does not directly accommodate the arbitrary-length tweaks required by NIST. For AE, the authors propose the dedicated construction aaa-bbb-ddd-AES instead of EtE. In aaa-bbb-ddd-AES, an independent universal hash function compresses the associated data and the portion of the nonce exceeding 96 bits into a τ-bit value, which is encoded in place of the 0^τ redundancy used in EtE.

The aaa-bbb-ddd-AES construction does not specify a concrete instantiation of this universal hash family, and therefore a precise security analysis is not possible without making an additional assumption. Since bbb-ddd-AES is designed to *"exclusively use similar building blocks as AES-GCM (AES and finite field multiplication)"* [56], we assume that the universal hash used to process the associated data is likewise instantiated with a polynomial hash.

Under this assumption, $\mathsf{CMT}^*_{\mathsf{k}}$ security can be broken with $O(1)$ cost. Specifically, one can choose two distinct polynomial-hash keys and efficiently compute common associated-data values that collide under the two keys, while keeping all other key components and the plaintext identical. The two resulting encryptions then produce the same ciphertext under distinct hash keys.

6 Efficient Key-Committing Transform for Accordions

We present the transform CACH that converts any accordion to a key-committing and robust-AE-secure scheme.

6.1 Primitives

Accordion. An accordion Acc is a set of variable-length permutations indexed by a key and a tweak. Let l be the key length of the accordion, and $\mathcal{T}$ be the the tweak set of Acc. The encryption with a key $L \in \mathcal{L}$ is denoted by Acc_L and the decryption is denoted by Acc_L^{-1}.

Algorithm 4 $\mathsf{CMT_k}$ Transform CACH

Encryption $\mathsf{CACH}[\mathsf{Acc}, \mathsf{CTR}[E_K], \mathsf{CEtH}[E_K]]((T, M)$

1: $L \leftarrow \mathsf{CTR}[E_K](\varepsilon)$; $C \leftarrow \mathsf{Acc}_L(T, M)$; $S \leftarrow \mathsf{msb}_{2n}(M \oplus C)$; $(S'_1, S'_2) \xleftarrow{n} S$
2: $Z \leftarrow \mathsf{CEtH}_K(\mathsf{msb}_{n-3}(S'_1), \mathsf{msb}_{n-2}(S'_2))$; **return** (C, Z)

Decryption $\mathsf{CACH}^{-1}[\mathsf{Acc}^{-1}, \mathsf{CTR}[E_K], \mathsf{CEtH}[E_K]](T, M)$

1: $L \leftarrow \mathsf{CTR}[E_K](\varepsilon)$; $M \leftarrow \mathsf{Acc}_L^{-1}(T, C)$; $S \leftarrow \mathsf{msb}_{2n}(M \oplus C)$; $(S'_1, S'_2) \xleftarrow{n} S$
2: $Z \leftarrow \mathsf{CEtH}[E_K](S_1, S_2)$; **return** (M, Z)

Subroutine $\mathsf{CTR}[E_K](\varepsilon)$

1: $L \leftarrow \mathsf{msb}_l(E_K(\mathsf{str}_{n-2}(0)\|00)\|E_K(\mathsf{str}_{n-2}(1)\|00)\|\cdots\|E_K(\mathsf{str}_{n-2}(\lceil l/n-1\rceil)\|00))$
2: **return** L

Subroutine $\mathsf{CEtH}_K(S_1, S_2)$

1: $\dot{L} \leftarrow (E_K(S_2\|01) \oplus E_K(S_2\|10))\|(E_K(S_2\|01) \oplus E_K(S_2\|11))$ ▷ CENC
2: $(\ddot{L}, \ddot{L}_+) \xleftarrow{k} \dot{L}\|0^{k-n}$
3: $Z_1 \leftarrow E_{\ddot{L}}(S_1\|000) \oplus S_1\|000$; $Z_2 \leftarrow E_{\ddot{L}}(S_1\|100) \oplus (S_1\|100)$ ▷ Hirose
4: $Z_1 \leftarrow Z_1 \oplus \ddot{L}_+$; $Z \leftarrow \mathsf{msb}_t(Z_1\|Z_2)$; **return** Z

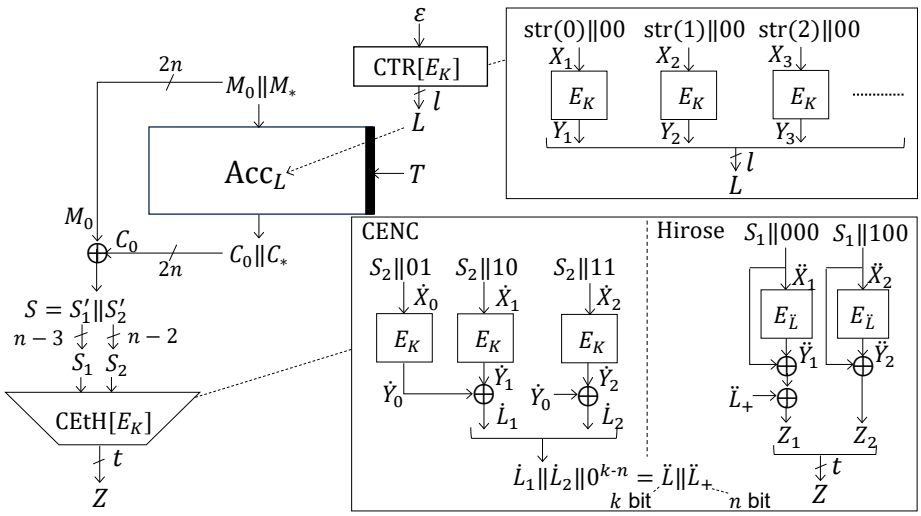

Fig. 10. CACH Transform. $|S'_1| = |S'_2| = n$, $S_1 = \mathsf{msb}_{n-3}(S'_1)$, and $S_2 = \mathsf{msb}_{n-2}(S'_2)$.

Block Cipher (BC). A BC is a set of permutations indexed by a key. For positive integers k and n, let $E : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ be an encryption of a BC with k-bit keys and n-bit blocks. Let $E^{-1} : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ be its decryption. Let $E^{\pm} := (E, E^{-1})$. Let $\mathcal{BC}_{k,n}$ be the set of all encryptions of k-bit key and n-bit block BCs. E with a key K is denoted by E_K or $E(K, \cdot)$. Similarly, E^{-1} with a key K is denoted by E_K^{-1} or $E^{-1}(K, \cdot)$.

6.2 CACH: Key-Committing Transform for Accordion

Design Strategy. CACH adds key commitment to an accordion without modifying its internal structure. It derives the accordion key from a master key K using the CTR-based key derivation function and computes an additional tag using CEtH. The central component is CEtH, which must provide two complementary properties: resistance to collisions across adversarially chosen keys for CMT_k security, and pseudorandomness on the inputs supplied by the accordion for mu-RAE security.

To obtain randomness in the tag input for both encryption and decryption, we derive this input from the most significant $2n$ bits of $M \oplus C$, where M and C are the plaintext and ciphertext of the underlying accordion. In the security analysis, replacing the accordion with random-bit oracles makes C fresh and uniform for encryption queries and M fresh and uniform for decryption queries. Thus, $M \oplus C$ supplies randomness in either direction. Using approximately $2n$ bits, rather than n bits, avoids an n-bit birthday bottleneck caused by repetitions of the tag input. In the concrete construction, the two input parts S_1 and S_2 together contain $2n - 5$ bits, with five bit positions reserved for domain separation.

The design of CEtH combines the collision-resistance intuition of the Merkle-Damgård (MD) paradigm [17,48] with the PRF-based key derivation used in the tweakable BC TDR [49]. Specifically, CEtH first processes S_2 using CENC [40] to generate $2n$-bit key material, and then processes S_1 using a Hirose-based compression stage [34]. CENC provides pseudorandom key material in the secret-key setting and serves as the first stage of our cross-key collision-resistance analysis. Its output supplies the BC key for the Hirose stage; when $k < 2n$, the remaining bits are incorporated through an additional feed-forward XOR. The resulting t-bit output forms the tag.

This CENC-then-Hirose structure follows the MD-inspired strategy of processing the input in stages while maintaining collision resistance. The TDR-inspired key-derivation step, together with the randomness supplied by the accordion, underlies the mu-RAE analysis. Sections 6.4 and 6.5 give the resulting bounds: BBB mu-RAE security under the stated conditions and $t/2$-bit CMT_k security in the ideal-cipher model.

Specification. CACH is specified in Algorithm 4 and illustrated in Fig. 10. It uses the n-bit BC E with k-bit keys, where $n \le k \le 2n$, and an accordion Acc with l-bit keys. The key, tweak, and plaintext spaces of CACH are $\{0,1\}^k$, $\mathcal{T}$, and $\{0,1\}^{\ge 2n}$, respectively. Thus, all plaintexts are at least $2n$ bits long. Let t denote the tag length in bits.

Encryption takes a key $K \in \{0,1\}^k$, a tweak $T \in \mathcal{T}$, and a plaintext $M \in \{0,1\}^{\ge 2n}$, and returns a ciphertext $C \in \{0,1\}^{|M|}$ and a tag $Z \in \{0,1\}^t$. It first derives the accordion key $L \in \{0,1\}^l$ as $L \leftarrow \mathsf{CTR}[E_K](\varepsilon)$, where $\mathsf{CTR}[E_K]$ is a BC-based key derivation function instantiated using counter mode. Since its input is fixed to the empty string ε, the key L depends only on K. It then computes the ciphertext as $C \leftarrow \mathsf{Acc}_L(T, M)$. To generate the tag, it sets S to the most significant $2n$ bits of $M \oplus C$ and splits S into two n-bit strings S'_1 and

S_2'. It then defines S_1 and S_2 as the most significant $n-3$ and $n-2$ bits of S_1' and S_2', respectively, and computes $Z \leftarrow \mathsf{CEtH}[E_K](S_1, S_2)$.

The collision-resistant PRF $\mathsf{CEtH}[E_K]$ consists of a CENC part followed by a part based on Hirose's compression function Hirose. The CENC part takes the $(n-2)$-bit input S_2 and uses E_K to derive a $2n$-bit value $\dot{L}$. The Hirose part processes the $(n-3)$-bit input S_1 using the BC $E_{\ddot{L}}$, where $\ddot{L}$ consists of the first k bits of $\dot{L}$. The remaining $2n-k$ bits of $\dot{L}$ are padded on the right with zeros to form a n-bit value, which is additionally XORed into the first output block of the Hirose part. The tag Z consists of the most significant t bits of the concatenation of the two resulting output blocks.

Decryption CACH^{-1} takes a key $K \in \{0,1\}^k$, a tweak $T \in \mathcal{T}$, and a ciphertext $C \in \{0,1\}^{\geq 2n}$, and returns a plaintext $M \in \{0,1\}^{|C|}$ and a tag $Z \in \{0,1\}^t$. It derives L as in encryption and computes $M \leftarrow \mathsf{Acc}_L^{-1}(T, C)$. It then computes Z from M and C using the same tag-generation procedure as encryption.

6.3 Security Definitions

Ideal Cipher Model. In our proofs, we model the n-bit BC with k-bit keys as an ideal cipher (IC) by sampling $E \xleftarrow{\$} \mathcal{BC}_{k,n}$. In the IC model, all parties, including the AE oracles and the adversary, access the BC through the shared oracle pair $E^{\pm} = (E, E^{-1})$. We call a direct query by the adversary to $E^{\pm}$ an offline query, and a query to a keyed symmetric-key scheme an online query.

Mu-RAE Security. We use the mu-RAE-security notion defined in [51]. Let $\Sigma_K \coloneqq \mathsf{CACH}[\mathsf{Acc}, \mathsf{CTR}[E_K], \mathsf{CEtH}[E_K]]$ and $\Sigma_K^{-1} \coloneqq \mathsf{CACH}^{-1}[\mathsf{Acc}^{-1}, \mathsf{CTR}[E_K], \mathsf{CEtH}[E_K]]$. The mu-RAE notion captures indistinguishability between (Σ, Σ^{-1}) and an ideal AE with decryption leakage in the multi-user setting.[6]

In the mu-RAE-security game with u users, an adversary $\mathbf{A}$ has access to either the real-world oracles $\mathcal{O}_{\mathsf{real}} \coloneqq ((\Sigma_{K^{[\nu]}}, \Sigma_{K^{[\nu]}}^{-1})_{\nu \in [u]}, E^{\pm})$ or the ideal-world oracles $\mathcal{O}_{\mathsf{ideal}} \coloneqq ((\$_{\mathsf{enc}}^{[\nu]}, \$_{\mathsf{dec}}^{[\nu]})_{\nu \in [u]}, E^{\pm})$, where $E^{\pm}$ is an IC. For each $\nu \in [u]$, the key of user ν is sampled as $K^{[\nu]} \xleftarrow{\$} \mathcal{K}$, where $\mathcal{K} = \{0,1\}^k$, independently of the other users' keys and the IC. The random-bit encryption oracle $\$_{\mathsf{enc}}^{[\nu]}$ takes a tweak–plaintext pair (T, M) and returns a ciphertext–tag pair sampled as $(C, Z) \xleftarrow{\$} \{0,1\}^{|M|} \times \{0,1\}^t$. Similarly, the random-bit decryption oracle $\$_{\mathsf{dec}}^{[\nu]}$ takes a tweak–ciphertext pair (T, C) and returns a plaintext–tag pair sampled as $(M, Z) \xleftarrow{\$} \{0,1\}^{|C|} \times \{0,1\}^t$. Each response is sampled independently of all previous responses. The adversary makes no repeated online query to the same oracle. Moreover, after obtaining an encryption (resp. decryption) query–response tuple (T, M, C, Z), it is forbidden to make the trivial decryption (resp. encryption) query (T, C) (resp. (T, M)) to the same user. At the end of the game, $\mathbf{A}$ outputs a decision bit in $\{0,1\}$. Let $\mathbf{A}^{\mathcal{O}} \in \{0,1\}$ denote the output of $\mathbf{A}$ with access

[6] The ideal AE considered here samples its responses with replacement, whereas the original notion uses sampling without replacement.

to a collection of oracles $\mathcal{O}$. The mu-RAE-security advantage of $\mathbf{A}$ is defined as $\mathbf{Adv}^{\mathsf{mu\text{-}rae}}_{\mathsf{CACH}}(\mathbf{A}) \coloneqq \Pr[\mathbf{A}^{\mathcal{O}_{\mathsf{real}}} = 1] - \Pr[\mathbf{A}^{\mathcal{O}_{\mathsf{ideal}}} = 1]$.

Assumption for the Accordion. To establish the mu-RAE security of CACH, we assume multi-user variable-input-length tweakable SPRP (mu-VIL-STPRP) security of the underlying accordion Acc. Let u be the number of users, and let Acc_L and Acc_L^{-1} denote accordion encryption and decryption under a key $L \in \{0,1\}^l$, respectively. Let $\widetilde{\mathsf{WPerm}}$ be the set of all tweakable, length-preserving permutations with the same interface as Acc_L. In the mu-VIL-STPRP-security game, an adversary $\mathbf{A}$ interacts with either the real-world oracles $\mathcal{O}_{\mathsf{real}} \coloneqq (\mathsf{Acc}_{L^{[\nu]}}, \mathsf{Acc}^{-1}_{L^{[\nu]}})_{\nu\in[u]}$ or the ideal-world oracles $\mathcal{O}_{\mathsf{ideal}} \coloneqq (\widetilde{\Psi}_\nu, \widetilde{\Psi}_\nu^{-1})_{\nu\in[u]}$. For each $\nu \in [u]$, the key $L^{[\nu]} \xleftarrow{\$} \{0,1\}^l$ is sampled independently in the real world, and the ideal accordion $\widetilde{\Psi}_\nu \xleftarrow{\$} \widetilde{\mathsf{WPerm}}$ is sampled independently in the ideal world. At the end of the game, $\mathbf{A}$ outputs a decision bit in $\{0,1\}$. Its mu-VIL-STPRP advantage is defined as $\mathbf{Adv}^{\mathsf{mu\text{-}vil\text{-}stprp}}_{\mathsf{Acc}}(\mathbf{A}) \coloneqq \Pr[\mathbf{A}^{\mathcal{O}_{\mathsf{real}}} = 1] - \Pr[\mathbf{A}^{\mathcal{O}_{\mathsf{ideal}}} = 1]$.

6.4 Mu-RAE Security of CACH

The following theorem establishes the mu-RAE security of CACH in the IC model. The proof is given in Section 7.

Theorem 1. *Let $\mathbf{A}$ be an adversary running in time t and making at most p offline queries and at most q online queries across u users. Suppose that the total length of the online queries, measured in n-bit blocks, is at most σ, and that at most q_{u} online queries are made to any single user. Then there exists an adversary $\mathbf{B}$ such that*

$$\mathbf{Adv}^{\mathsf{mu\text{-}rae}}_{\mathsf{CACH}}(\mathbf{A}) \le \frac{3q_{\mathsf{u}}^2 q}{2^{2n}} + \frac{1.5uq}{2^k} + \left(u + \frac{2n}{\log_2 n}\right)\cdot\frac{p}{2^k} + \frac{0.5(\lceil l/n\rceil)^2 u}{2^n} + \left(\frac{60(\log_2 n)q}{2^n}\right)^{\frac{n}{\log_2 n}} + \frac{q}{2^n} + \frac{33q_{\mathsf{u}}q}{2^{2n}} + \frac{18q^2}{2^{k+n}} + \mathbf{Adv}^{\mathsf{mu\text{-}vil\text{-}stprp}}_{\mathsf{Acc}}(\mathbf{B}),$$

where $\mathbf{B}$ runs in time $t + O(\sigma + p)$ and makes at most q queries to at most u users, with total query length at most σ in n-bit blocks.

Assume that the mu-VIL-STPRP advantage of Acc is negligible compared with the other terms. Then, the first three terms determine the relevant query limits. The first two terms concern online queries and are negligible provided that $q \ll \min\{2^{2n}/q_{\mathsf{u}}^2, 2^k/u\}$. Considering these two terms, the online security level is $\min\{2n - 2\log_2 q_{\mathsf{u}}, k - \log_2 u\}$ bits. In particular, when $q \approx q_{\mathsf{u}}$ and $u \le 2^{k-2n/3}$, this gives $2n/3$-bit online security. The third term concerns offline queries and is negligible provided that $p \ll 2^k/u$. This corresponds to an offline security level of $k - \log_2 u$ bits. The resulting query threshold matches that of generic multi-user key recovery, $2^k/u$.

6.5 $\mathsf{CMT_k}$ Security of CACH

The following theorem establishes the $\mathrm{CMT_k}$ security of CACH in the IC model, giving $t/2$-bit security.

Theorem 2. *Assume that $n \le t \le 2n$, $n \le k \le 2n$, and $3p < 2^n$. For any adversary $\mathbf{A}$ making at most p queries to the IC $E^{\pm}$, we have $\mathrm{Adv}^{\mathrm{cmt_k}}_{\mathsf{CACH}}(\mathbf{A}) \le \epsilon$, where $\epsilon = 2^{2n-t}p^2/(2^n-3p)^2$ if $k = 2n$, and $\epsilon = 2^{2n-t}\left(\frac{n}{\log_2 n}\cdot\frac{p}{2^n-3p}\right)^2 + \left(6(\log_2 n)\cdot\frac{p}{2^n-3p}\right)^{\frac{n}{\log_2 n}}$ if $n \le k < 2n$.*

The proof is given in Supplementary Material B. We outline its main ideas here. The $\mathrm{CMT_k}$ security of CACH reduces to the collision resistance of CEtH, which has a CENC-then-Hirose structure. For $p \ll 2^n$, the Hirose part has a birthday-bound collision probability of $O(p^2/2^t)$. The CENC part can be viewed as a Hirose-like construction with a keyed BC in the feed-forward path. Since this BC is a n-bit permutation, a similar evaluation gives a collision bound of $O(p^2/2^{2n})$ for the CENC part. When $k < 2n$, the additional feed-forward XOR in CEtH requires a further multi-collision evaluation. Combining these evaluations gives the stated bound.

7 Proof of Theorem 1

Without loss of generality, we assume that the adversary $\mathbf{A}$ is deterministic and makes no repeated queries.

Definitions used in the Proof. In our proofs, we use the following notation and definitions.

- For $\nu \in [u]$ and $\alpha \in [q]$, ν is called a "user index" and α is called a "query index."
- For $\alpha \in [q]$, values corresponding to the α-th online query are denoted using the superscript (α), such as $M^{(\alpha)}$, $C^{(\alpha)}$, etc.
- For $\alpha \in [p]$, the query-response tuple of the α-th offline query is denoted by $(\hat{K}^{(\alpha)}, \hat{X}^{(\alpha)}, \hat{Y}^{(\alpha)})$, where $\hat{Y}^{(\alpha)} = E_{\hat{K}^{(\alpha)}}(\hat{X}^{(\alpha)})$.
- For $\alpha \in [q]$, let $\mathsf{u}_\alpha \in [u]$ be the user index for the α-th query. If the α-th query is made to the ν-th user, then $\mathsf{u}_\alpha = \nu$.
- Let $\mathcal{Q}^{[\nu]} \subseteq [q]$ be the set of query indices for the ν-th user.
- We call "a query phase" a phase in which an adversary makes queries to its oracles and "a decision phase" a phase after all queries have been completed and before outputting a decision bit.

Security Games. In this proof, we consider the following seven games. For $i \in [7]$, let $\mathcal{O}_i$ be a collection of oracles in Game i. In Game i, $\mathbf{A}$ has access to the oracles in $\mathcal{O}_i$. For $\nu \in [u]$, let $\mathcal{O}_i^{[\nu]}$ be the oracle for the ν-th user in Game i. Hence, $\mathcal{O}_i = (\mathcal{O}_i^{[1]}, \ldots, \mathcal{O}_i^{[u]}, E^{\pm})$. These oracles are defined as follows.

– Game 1: This corresponds to the real world. Hence, $\mathcal{O}_1$ is defined as follows.

$$\forall \nu \in [u] : \mathcal{O}_1^{[\nu]} \coloneqq (\mathsf{CACH}[\mathsf{Acc}, \mathsf{CTR}[E_{K^{[\nu]}}], \mathsf{CEtH}[E_{K^{[\nu]}}]], \\ \mathsf{CACH}^{-1}[\mathsf{Acc}^{-1}, \mathsf{CTR}[E_{K^{[\nu]}}], \mathsf{CEtH}[E_{K^{[\nu]}}]]),$$

where the initializations $K^{[\nu]} \xleftarrow{\$} \{0,1\}^k$ and $E \xleftarrow{\$} \mathcal{BC}_{k,n}$ are performed before the first query.

– Game 2: From Game 1 to Game 2, the BCs $(E_{K^{[\nu]}})_{\nu \in [u]}$ (except for the Hirose part) are replaced with random permutations (RPs) $(\pi_\nu)_{\nu \in [u]}$. Here, $\mathsf{CEtH}[\pi_\nu]$ denotes the construction obtained by replacing $E_{K^{[\nu]}}$ with π_ν only in the CENC part, while the Hirose part continues to use the shared IC E. Let Perm_n be the set of n-bit permutations. $\mathcal{O}_2$ is defined as follows.

$$\forall \nu \in [u] : \mathcal{O}_2^{[\nu]} \coloneqq (\mathsf{CACH}[\mathsf{Acc}, \mathsf{CTR}[\pi_\nu], \mathsf{CEtH}[\pi_\nu]], \\ \mathsf{CACH}^{-1}[\mathsf{Acc}^{-1}, \mathsf{CTR}[\pi_\nu], \mathsf{CEtH}[\pi_\nu]]),$$

where the initializations $E \xleftarrow{\$} \mathcal{BC}_{k,n}$ and $\forall \nu \in [u] : \pi_\nu \xleftarrow{\$} \mathsf{Perm}_n$ are performed before the first query.

– Game 3: From Game 2 to Game 3, the functions $(\mathsf{CTR}[\pi_\nu])_{\nu \in [u]}$ are replaced with RFs $(\mathcal{F}_{\mathsf{CTR}}^{[\nu]})_{\nu \in [u]}$. Let $\mathsf{Func}_{\mathsf{CTR}}$ be the set of functions with the same interface as $\mathsf{CTR}[E_{K^{[\nu]}}]$. $\mathcal{O}_3$ is defined as follows.

$$\forall \nu \in [u] : \mathcal{O}_3^{[\nu]} \coloneqq (\mathsf{CACH}[\mathsf{Acc}, \mathcal{F}_{\mathsf{CTR}}^{[\nu]}, \mathsf{CEtH}[\pi_\nu]], \\ \mathsf{CACH}^{-1}[\mathsf{Acc}^{-1}, \mathcal{F}_{\mathsf{CTR}}^{[\nu]}, \mathsf{CEtH}[\pi_\nu]]),$$

where the initializations $E \xleftarrow{\$} \mathcal{BC}_{k,n}$ and $\forall \nu : \pi_\nu \xleftarrow{\$} \mathsf{Perm}_n; \mathcal{F}_{\mathsf{CTR}}^{[\nu]} \xleftarrow{\$} \mathsf{Func}_{\mathsf{CTR}}$ are performed before the first query.

– Game 4: From Game 3 to Game 4, the accordions $(\mathsf{Acc}_{L^{[\nu]}})_{\nu \in [u]}$ such that $L^{[\nu]} = \mathcal{F}_{\mathsf{CTR}}^{[\nu]}(\varepsilon)$ are replaced with ideal accordions $(\widetilde{\Psi}_\nu^{\pm})_{\nu \in [u]}$. Here, $\mathsf{CACH}[\widetilde{\Psi}_\nu, \mathsf{CEtH}[\pi_\nu]]$ denotes the construction obtained from CACH by directly using the given accordion oracle $\widetilde{\Psi}_\nu$ without deriving an accordion key via CTR. $\mathcal{O}_4$ is defined as follows. Note that in $\mathcal{O}_4^{[\nu]}$, the RF $\mathcal{F}_{\mathsf{CTR}}^{[\nu]}$ is not used and is therefore omitted.

$$\forall \nu \in [u] : \mathcal{O}_4^{[\nu]} \coloneqq (\mathsf{CACH}[\widetilde{\Psi}_\nu, \mathsf{CEtH}[\pi_\nu]], \mathsf{CACH}^{-1}[\widetilde{\Psi}_\nu^{-1}, \mathsf{CEtH}[\pi_\nu]]),$$

where the initializations $E \xleftarrow{\$} \mathcal{BC}_{k,n}$ and $\forall \nu : \pi_\nu \xleftarrow{\$} \mathsf{Perm}_n; \widetilde{\Psi}_\nu \xleftarrow{\$} \widetilde{\mathsf{WPerm}}$ are performed before the first query.

– Game 5: From Game 4 to Game 5, the ideal accordions $(\widetilde{\Psi}_\nu, \widetilde{\Psi}_\nu^{-1})_{\nu \in [u]}$ are replaced with random-bit oracles $(\$_{\mathsf{enc}}^{\prime[\nu]}, \$_{\mathsf{dec}}^{\prime[\nu]})_{\nu \in [u]}$, where $\$_{\mathsf{enc}}^{\prime[\nu]}$ (resp. $\$_{\mathsf{dec}}^{\prime[\nu]}$) returns a random string of length equal to $|\widetilde{\Psi}_\nu(T, M)|$ (resp. $|\widetilde{\Psi}_\nu(T, C)|$) for a query (T, M) (resp. (T, C)). $\mathcal{O}_5$ is defined as follows.

$$\forall \nu \in [u] : \mathcal{O}_5^{[\nu]} \coloneqq (\mathsf{CACH}[\$_{\mathsf{enc}}^{\prime[\nu]}, \mathsf{CEtH}[\pi_\nu]], \mathsf{CACH}^{-1}[\$_{\mathsf{dec}}^{\prime[\nu]}, \mathsf{CEtH}[\pi_\nu]]),$$

where the initializations $E \xleftarrow{\$} \mathcal{BC}_{k,n}$ and $\forall \nu : \pi_\nu \xleftarrow{\$} \mathsf{Perm}_n$ are performed before the first query.

– Game 6: From Game 5 to Game 6, the functions $(\mathsf{CEtH}[\pi_\nu])_{\nu\in[u]}$ are replaced with RFs $(\mathcal{F}^{[\nu]}_{\mathsf{CEtH}})_{\nu\in[u]}$. Let $\mathsf{Func}_{\mathsf{CEtH}}$ be the set of functions with the same interface as $\mathsf{CEtH}[\pi_\nu]$. $\mathcal{O}_6$ is defined as follows.

$$\forall \nu \in [u] : \mathcal{O}_6^{[\nu]} \coloneqq (\mathsf{CACH}[\$'^{[\nu]}_{\mathsf{enc}}, \mathcal{F}^{[\nu]}_{\mathsf{CEtH}}], \mathsf{CACH}^{-1}[\$'^{[\nu]}_{\mathsf{dec}}, \mathcal{F}^{[\nu]}_{\mathsf{CEtH}}]),$$

where the initializations $E \xleftarrow{\$} \mathcal{BC}_{k,n}$ and $\forall \nu : \mathcal{F}^{[\nu]}_{\mathsf{CEtH}} \xleftarrow{\$} \mathsf{Func}_{\mathsf{CEtH}}$ are performed before the first query.

– Game 7: Game 7 corresponds to the ideal world. $\mathcal{O}_7$ is defined as $\forall \nu \in [u] : \mathcal{O}_7^{[\nu]} \coloneqq (\$^{[\nu]}_{\mathsf{enc}}, \$^{[\nu]}_{\mathsf{dec}})$.

Using these games, we have $\mathbf{Adv}^{\mathsf{mu\text{-}rae}}_{\mathsf{CACH}}(\mathbf{A}) = \Pr[\mathbf{A}^{\mathcal{O}_1} = 1] - \Pr[\mathbf{A}^{\mathcal{O}_7} = 1] = \sum_{i\in[6]} \mathbf{Adv}^{\mathsf{dist}}_{\mathcal{O}_i,\mathcal{O}_{i+1}}(\mathbf{A})$. The bounds on these differences $(\mathbf{Adv}^{\mathsf{dist}}_{\mathcal{O}_i,\mathcal{O}_{i+1}}(\mathbf{A}))_{i\in[6]}$ are given in the following evaluations, yielding the bound in Theorem 1.

Deriving the Bound of $\mathbf{Adv}^{\mathsf{dist}}_{\mathcal{O}_1,\mathcal{O}_2}(\mathbf{A})$. We consider the following event for a collision with a users' key in Game 1.

– $\mathsf{coll}_K \Leftrightarrow (\exists \nu \in [u], \alpha \in [q] \text{ s.t. } K^{[\nu]} = \ddot{L}^{(\alpha)}) \vee (\exists \nu \in [u], \beta \in [p] \text{ s.t. } K^{[\nu]} = \hat{K}^{(\beta)}) \vee (\exists \{\nu_1, \nu_2\} \in \mathsf{UConf}_2([u]) \text{ s.t. } K^{[\nu_1]} = K^{[\nu_2]})$

For each $\nu \in [u]$, $K^{[\nu]}$ is chosen uniformly at random from $\{0,1\}^k$, thus we have $\Pr[\mathsf{coll}_K] \le \frac{u(q+p)}{2^k} + \frac{0.5u^2}{2^k}$. Assuming that coll_K does not occurs, for each $\nu \in [u]$, the output of $E_{K^{[\nu]}}$ is defined independently of the other IC queries. We thus have $\mathbf{Adv}^{\mathsf{dist}}_{\mathcal{O}_1,\mathcal{O}_2}(\mathbf{A}) \le \Pr[\mathsf{coll}_K] \le \frac{u(p+1.5q)}{2^k}$.

Deriving the Bound of $\mathbf{Adv}^{\mathsf{dist}}_{\mathcal{O}_2,\mathcal{O}_3}(\mathbf{A})$. From Game 2 to Game 3, the functions $(\mathsf{CTR}[\pi_\nu])_{\nu\in[u]}$ are replaced with RFs $(\mathcal{F}^{[\nu]}_{\mathsf{CTR}})_{\nu\in[u]}$. By the RP-RF switch, the difference between the RP's outputs and the PF's outputs is bounded by $\frac{0.5(\lceil l/n\rceil)^2 u}{2^n}$. We thus have $\mathbf{Adv}^{\mathsf{dist}}_{\mathcal{O}_2,\mathcal{O}_3}(\mathbf{A}) \le \frac{0.5(\lceil l/n\rceil)^2 u}{2^n}$.

Deriving the Bound of $\mathbf{Adv}^{\mathsf{dist}}_{\mathcal{O}_3,\mathcal{O}_4}(\mathbf{A})$. From Game 3 to Game 4, the accordions $(\mathsf{Acc}_{L^{[\nu]}})_{\nu\in[u]}$ such that $L^{[\nu]} = \mathcal{F}^{[\nu]}_{\mathsf{CTR}}(\varepsilon)$ are replaced with ideal accordions $(\widetilde{\Psi}_\nu, \widetilde{\Psi}_\nu^{-1})_{\nu\in[u]}$. Since each accordion's key is chosen uniformly at random from the key space, the difference can be bounded by the $\mathsf{mu\text{-}VIL\text{-}STPRP}$ advantage function for Acc as follows: there exists an adversary such that $\mathbf{Adv}^{\mathsf{dist}}_{\mathcal{O}_3,\mathcal{O}_4}(\mathbf{A}) \le \mathbf{Adv}^{\mathsf{mu\text{-}vil\text{-}stprp}}_{\mathsf{Acc}}(\mathbf{B})$, where $\mathbf{B}$ runs in time $t + O(\sigma + p)$ and makes q queries to at most u users such that the total block length in all queries is at most σ

Deriving the Bound of $\mathbf{Adv}^{\mathsf{dist}}_{\mathcal{O}_4,\mathcal{O}_5}(\mathbf{A})$. From Game 4 to Game 5, the ideal accordions $(\widetilde{\Psi}_\nu, \widetilde{\Psi}_\nu^{-1})_{\nu\in[u]}$ are replaced with random-bit oracles $(\$'^{[\nu]}_{\mathsf{enc}}, \$'^{[\nu]}_{\mathsf{dec}})_{\nu\in[u]}$. Since the plaintext length is $2n$ bits or grater, the RP-RF switching offers the following bound: $\mathbf{Adv}^{\mathsf{dist}}_{\mathcal{O}_4,\mathcal{O}_5}(\mathbf{A}) \le \sum_{\nu\in[\mathrm{u}]} \sum_{\{\alpha,\beta\}\in\mathsf{UConf}_2(\mathcal{Q}^{[\nu]})} \frac{1}{2^{2n}} \le \frac{0.5q_u q}{2^{2n}}$.

Deriving the Bound of $\mathbf{Adv}^{\mathsf{dist}}_{\mathcal{O}_5,\mathcal{O}_6}(\mathbf{A})$. We give an intuition of the evaluation. The detail evaluation is given in Supplementary Material A.

CEtH has the CENC-then-Hirose construction, where the BC key of Hirose is defined by using CENC. Regarding the CENC part, as the previous work [40], we can show that CENC achieves the beyond-birthday-bound PRF security in the multi-user setting. The bound is about $\frac{q_u^2 q}{2^{2n}}$. Note that the previous work only show the security in the single-user setting. Regarding the Hirose part, the structure can be seen as the beyond-birthday-bound secure tweakable BC TDR [49] where a BC's key is defined by using a PRF with a tweak. Moreover, the input to the Hirose part is defined by the output of a random-bit oracle. By combining the TDR-like evaluation with the randomness of the input to the Hirose part, we can shows outputs of the Hirose part is indistinguisbale from random values up to the bound $O(\frac{q}{2^n} + \frac{p}{2^k})$. Then, we obtain the following bound $\mathbf{Adv}^{\mathsf{dist}}_{\mathcal{O}_5,\mathcal{O}_6}(\mathbf{A}) \leq \frac{3q_u^2 q}{2^{2n}} + \frac{2n}{\log_2 n} \cdot \frac{p}{2^k} + \left(\frac{60(\log_2 n)q}{2^n}\right)^{\frac{n}{\log_2 n}} + \frac{q}{2^n} + \frac{16q_u q}{2^{2n}} + \frac{18q^2}{2^{k+n}}$.

Deriving the Bound of $\mathbf{Adv}^{\mathsf{dist}}_{\mathcal{O}_6,\mathcal{O}_7}(\mathbf{A})$. In Game 6, for each $\alpha \in [q]$, the output $Z^{(\alpha)}$ is chosen uniformly at random from $\{0,1\}^t$ as long as the following collision event does not occur for $S_1 \| S_2$ values in the same user.

- $\mathsf{coll}_S \Leftrightarrow \exists \nu \in [u], \alpha, \beta \in \mathcal{Q}^{[\nu]}$ s.t. $\alpha < \beta \wedge S_1^{(\alpha)} \| S_2^{(\alpha)} = S_1^{(\beta)} \| S_2^{(\beta)}$.

We then have $\mathbf{Adv}^{\mathsf{dist}}_{\mathcal{O}_6,\mathcal{O}_7}(\mathbf{A}) \leq \Pr[\mathsf{coll}_S]$. Since for each β, $M^{(\beta)}$ or $C^{(\beta)}$ is chosen uniformly at random from $\{0,1\}^{|M^{(\beta)}|}$, we have $\mathbf{Adv}^{\mathsf{dist}}_{\mathcal{O}_6,\mathcal{O}_7}(\mathbf{A}) \leq \Pr[\mathsf{coll}_S] \leq \sum_{\nu \in [u]} \sum_{\{\alpha,\beta\} \in \mathsf{UConf}_2(\mathcal{Q}^{[\nu]})} \frac{2^5}{2^{2n}} \leq \frac{16q_u q}{2^{2n}}$.

8 Software Performance Evaluation

We benchmark the software performance of CHCTR2 combined with either EtE or CACH, namely EtE-CHCTR2 and CACH-CHCTR2, respectively. These schemes are instantiated with AES-256. For comparison, we also implement the birthday-bound scheme EtE-HCTR2 instantiated with the 256-bit BC Rijndael-256, as considered by NIST [52]. All implementations are optimized using 128-bit SIMD instructions, including AES-NI [30] (see Appendix C for details).

Table 3 compares the performance of the target schemes in cycles per byte (cpb). This comparison aims to evaluate the overhead of CACH for short messages. First, the AES-256-based schemes (CACH-CHCTR2 and EtE-CHCTR2) are faster than the Rijndael-256-based scheme. This is largely because Rijndael-256 is less efficient than AES-256: realizing Rijndael-256 with AES-NI requires additional SIMD instructions (e.g., `vpshufb` and `vpblendvb`). Second, CACH-CHCTR2 is faster than EtE-CHCTR2 while achieving a higher security level. This is because the additional cost of CACH is smaller than the cost of processing the two extra message blocks introduced by the zero padding in EtE-CHCTR2. The additional cost of CACH is independent of the message size and is thus amortized as the message size increases, so that the performance of CACH-CHCTR2

Table 3. Cycles per byte (cpb) of CACH-CHCTR2, EtE-CHCTR2, and EtE-HCTR2 with short messages evaluated on the Intel Xeon w5-3423 (Sapphire Rapids) processor.

		Message size in bytes, excluding zero padding in EtE						
Scheme	BC	32	48	64	80	96	112	128
CACH-CHCTR2	AES-256	23.826	16.989	13.502	11.796	10.653	10.125	9.439
EtE-CHCTR2	AES-256	25.240	18.393	15.041	12.966	11.582	11.579	10.553
EtE-HCTR2	Rijndael-256	31.789	22.963	17.466	15.645	13.243	12.812	11.368

eventually converges to that of CHCTR2. Appendix C provides the results for larger messages.

References

[1] Albertini, A., Duong, T., Gueron, S., Kölbl, S., Luykx, A., Schmieg, S.: How to Abuse and Fix Authenticated Encryption Without Key Commitment. In: USENIX Security 2022. pp. 3291–3308 (2022)

[2] Andreeva, E., Bhati, A.S., Preneel, B., Vizár, D.: 1, 2, 3, Fork: Counter Mode Variants based on a Generalized Forkcipher. IACR Trans. Symmetric Cryptol. **2021**(3), 1–35 (2021)

[3] Andreeva, E., Bogdanov, A., Luykx, A., Mennink, B., Mouha, N., Yasuda, K.: How to Securely Release Unverified Plaintext in Authenticated Encryption. In: ASIACRYPT 2014. LNCS, vol. 8873, pp. 105–125 (2014)

[4] Android Open Source Project: File-based encryption. `https://source.android.com/docs/security/features/encryption/file-based` (2023)

[5] AWS Key Management Service: AWS KMS concepts. `https://docs.aws.amazon.com/kms/latest/developerguide/concepts.html` (Accessed on 09/17/2026)

[6] Bellare, M., Hoang, V.T.: Efficient Schemes for Committing Authenticated Encryption. In: EUROCRYPT 2022, Part II. LNCS, vol. 13276, pp. 845–875. Springer (2022)

[7] Bellare, M., Rogaway, P.: Encode-Then-Encipher Encryption: How to Exploit Nonces or Redundancy in Plaintexts for Efficient Cryptography. In: ASIACRYPT 2000. LNCS, vol. 1976, pp. 317–330. Springer (2000)

[8] Bernstein, D.J.: ChaCha, a variant of Salsa20. `https://cr.yp.to/chacha/chacha-20080128.pdf` (January 2008)

[9] Chang, D., Chen, Y.L., Hiraga, Y., Minematsu, K., Mouha, N., Naito, Y., Sasaki, Y., Shiba, R., Sugawara, T.: Extended Analysis of Key Committing Security of HCTR2. Preproceedings of SAC 2026 (2026), `https://sacworkshop.org/SAC26/accepted-papers.html`

[10] Chang, D., Chen, Y.L., Hiraga, Y., Minematsu, K., Mouha, N., Naito, Y., Sasaki, Y., Sugawara, T.: Key Committing Security of HCTR2, Revisited. In: CRYPTO 2026, Part VI. LNCS, vol. 16805, pp. 98–130. Springer (2026)

[11] Chen, Y.L., Davidson, M., Dworkin, M., Kelsey, J., Sasaki, Y., Turan, M.S., Chang, D., Mouha, N., Thompson, A.: Requirements for Cryptographic Accordions. NIST Interagency Report, NIST IR 8552 (Apr 2025), `https://csrc.nist.gov/pubs/ir/8552/final`

[12] Chen, Y.L., Flórez-Gutiérrez, A., Inoue, A., Ito, R., Iwata, T., Minematsu, K., Mouha, N., Naito, Y., Sibleyras, F., Todo, Y.: Key Committing Security of AEZ and More. IACR Trans. Symmetric Cryptol. **2023**(4), 452–488 (2023)

[13] Chen, Y.L., Hiraga, Y., Mouha, N., Naito, Y., Sasaki, Y., Sugawara, T.: Beyond-Birthday-Bound Security with HCTR2: Cascaded Construction and Tweak-Based Key Derivation. In: ASIACRYPT 2025, Part I. LNCS, vol. 16245, pp. 3–34. Springer (2025)
[14] Chung, W., Hwang, S., Kim, S., Lee, B., Lee, J.: Making GCM Great Again: Toward Full Security and Longer Nonces. In: EUROCRYPT 2025, Part I. LNCS, vol. 15601, pp. 33–61. Springer (2025)
[15] Crowley, P., Biggers, E.: Adiantum: length-preserving encryption for entry-level processors. IACR Trans. Symmetric Cryptol. **2018**(4), 39–61 (2018)
[16] Crowley, P., Huckleberry, N., Biggers, E.: Length-preserving encryption with HCTR2. IACR Cryptol. ePrint Arch. (2021), https://eprint.iacr.org/2021/1441
[17] Damgård, I.: A Design Principle for Hash Functions. In: CRYPTO '89. LNCS, vol. 435, pp. 416–427. Springer (1989)
[18] Datta, N., Degabriele, J.P., Dutta, A., Karadzic, V., Nandi, H.: Rugged Pseudorandom Permutations with Beyond-Birthday-Bound Security. In: ASIA CCS 2026. pp. 502–518. ACM (2026)
[19] Datta, N., Dutta, A., Ghosh, S., List, E., Nandi, H.: HCTR+: an optimally secure tbc-based accordion mode. IACR Trans. Symmetric Cryptol. **2025**(3), 183–229 (2025)
[20] Degabriele, J.P., Vukašin Karadžić: Overloading the Nonce: Rugged PRPs, Nonce-Set AEAD, and Order-Resilient Channels. In: Advances in Cryptology - CRYPTO 2022, Part IV. LNCS, vol. 13510, pp. 264–295. Springer (2022)
[21] Dobraunig, C., Matusiewicz, K., Mennink, B., Tereschenko, A.: Efficient Instances of Docked Double Decker with AES, and Application to Authenticated Encryption. In: EUROCRYPT 2025, Part I. LNCS, vol. 15601, pp. 62–92. Springer (2025)
[22] Dodis, Y., Grubbs, P., Ristenpart, T., Woodage, J.: Fast Message Franking: From Invisible Salamanders to Encryptment. In: CRYPTO 2018, Part I. LNCS, vol. 10991, pp. 155–186. Springer (2018)
[23] Drucker, N., Gueron, S.: Software optimization of Rijndael for modern x86-64 platforms. In: ITNG 2022 19th International Conference on Information Technology-New Generations. pp. 147–153 (2022)
[24] Dutta, A., Nandi, M.: Tweakable HCTR: A BBB Secure Tweakable Enciphering Scheme. In: INDOCRYPT 2018. LNCS, vol. 11356, pp. 47–69. Springer (2018)
[25] Dworkin, M.: NIST Special Publication 800-38A: Recommendation for block cipher modes of operation: Methods and techniques. https://csrc.nist.gov/pubs/sp/800/38/a/final (2001)
[26] Dworkin, M.: NIST Special Publication 800-38C: Recommendation for block cipher modes of operation: the CCM mode for authentication and confidentiality. https://csrc.nist.gov/pubs/sp/800/38/c/upd1/final (2007)
[27] Dworkin, M.: NIST Special Publication 800-38D: Recommendation for block cipher modes of operation: Galois/counter mode (GCM) and GMAC. https://csrc.nist.gov/pubs/sp/800/38/d/final (2007)
[28] Farshim, P., Orlandi, C., Rosie, R.: Security of symmetric primitives under incorrect usage of keys. IACR Trans. Symmetric Cryptol. **2017**(1), 449–473 (2017)
[29] Grubbs, P., Lu, J., Ristenpart, T.: Message Franking via Committing Authenticated Encryption. In: CRYPTO 2017, Part III. LNCS, vol. 10403, pp. 66–97. Springer (2017)
[30] Gueron, S.: Intel advanced encryption standard (AES) new instructions set. https://www.intel.com/content/dam/doc/white-paper/advanced-encryption-standard-new-instructions-set-paper.pdf (2010)

[31] Gueron, S., Kounavis, M.E.: Intel carry-less multiplication instruction and its usage for computing the GCM mode. https://www.intel.com/content/dam/develop/external/us/en/documents/clmul-wp-rev-2-02-2014-04-20.pdf (2014)
[32] Gueron, S., Langley, A., Lindell, Y.: AES-GCM-SIV: Nonce Misuse-Resistant Authenticated Encryption. RFC **8452**, 1–42 (2019)
[33] Halevi, S., Rogaway, P.: A Parallelizable Enciphering Mode. In: CT-RSA 2004. LNCS, vol. 2964, pp. 292–304. Springer (2004)
[34] Hirose, S.: Some Plausible Constructions of Double-Block-Length Hash Functions. In: FSE 2006. LNCS, vol. 4047, pp. 210–225. Springer (2006)
[35] Hirose, S., Minematsu, K.: Envelope encryption in the symmetric-key setting: A formalization and generic constructions. Cryptology ePrint Archive, Paper 2023/1727 (2023), https://eprint.iacr.org/2023/1727
[36] Hirose, S., Minematsu, K.: A Formal Treatment of Envelope Encryption. In: ICISC 2024. LNCS, vol. 15596, pp. 96–110. Springer (2024)
[37] Hoang, V.T., Krovetz, T., Rogaway, P.: Robust Authenticated-Encryption AEZ and the Problem That It Solves. In: EUROCRYPT 2015. LNCS, vol. 9056, pp. 15–44 (2015)
[38] Hoang, V.T., Menda, S.: Robust AE With Committing Security. In: ASIACRYPT 2024, Part IX. LNCS, vol. 15492, pp. 312–342. Springer (2024)
[39] Isobe, T., Ito, R., Minematsu, K.: Security Analysis of SFrame. In: ESORICS 2021, Part II. LNCS, vol. 12973, pp. 127–146. Springer (2021)
[40] Iwata, T.: New Blockcipher Modes of Operation with Beyond the Birthday Bound Security. In: FSE 2006. LNCS, vol. 4047, pp. 310–327. Springer (2006)
[41] Jankowski, K., Laurent, P.: Packed AES-GCM Algorithm Suitable for AES/PCLMULQDQ Instructions. IEEE Trans. Computers **60**(1), 135–138 (2011)
[42] Kampanakis, P., Campagna, M., Crocket, E., Petcher, A.: Practical Challenges with AES-GCM and the Need for a New Mode and Wide-Block Cipher. presented at NIST The Third NIST Workshop on Block Cipher Modes of Operation 2023, https://csrc.nist.gov/Presentations/2023/practical-challenges-with-aes-gcm (2023)
[43] Kerrisk, M.: perf_event_open(2) — Linux manual page. https://man7.org/linux/man-pages/man2/perf_event_open.2.html (2026)
[44] Krovetz, T., Rogaway, P.: The OCB Authenticated-Encryption Algorithm. RFC **7253**, 1–19 (2014). https://doi.org/10.17487/RFC7253
[45] Lee, B.: A BBB Secure Accordion Mode from HCTR. Workshop Presentation (Jun 2024), https://csrc.nist.gov/csrc/media/Presentations/2024/a-bbb-secure-accordion-mode-from-hctr/images-media/sess-8-lee-acm-workshop-2024.pdf
[46] Len, J., Grubbs, P., Ristenpart, T.: Partitioning Oracle Attacks. In: USENIX Security 2021. pp. 195–212 (2021)
[47] Menda, S., Len, J., Grubbs, P., Ristenpart, T.: Context Discovery and Commitment Attacks - How to Break CCM, EAX, SIV, and More. In: EUROCRYPT 2023, Part IV. pp. 379–407. LNCS, Springer (2023)
[48] Merkle, R.: One Way Hash Functions and DES. In: CRYPTO'89. LNCS, vol. 435, pp. 428–446. Springer (1990)
[49] Minematsu, K.: Beyond-Birthday-Bound Security Based on Tweakable Block Cipher. In: FSE 2009. LNCS, vol. 5665, pp. 308–326. Springer (2009)
[50] Naito, Y.: CHCTR Family for BBBAcc. Pre-Draft Public Comments on SP 800-197A . https://csrc.nist.gov/pubs/sp/800/197/a/iprd (August 2025)

[51] Naito, Y., Sasaki, Y., Sugawara, T.: Committing Wide Encryption Mode with Minimum Ciphertext Expansion. IACR Trans. Symmetric Cryptol. **2025**(1), 44–69 (2025)
[52] NIST: PRE-DRAFT Call for Comments: NIST Launches Development of Cryptographic Accordions. https://csrc.nist.gov/pubs/sp/800/197/a/iprd (2025)
[53] Patarin, J.: The "Coefficients H" Technique. In: Selected Areas in Cryptography, SAC 2008. LNCS, vol. 5381, pp. 328–345. Springer (2008)
[54] Rogaway, P.: Authenticated-Encryption with Associated-Data. In: Atluri, V. (ed.) Proceedings of the 9th ACM Conference on Computer and Communications Security, CCS 2002, Washington, DC, USA, November 18-22, 2002. pp. 98–107. ACM (2002)
[55] Rogaway, P., Bellare, M., Black, J., Krovetz, T.: OCB: A Block-Cipher Mode of Operation for Efficient Authenticated Encryption. In: CCS 2001, Proceedings of the 8th ACM Conference on Computer and Communications Security, Philadelphia, Pennsylvania, USA, November 6-8, 2001. pp. 196–205. ACM (2001)
[56] Shrimpton, T., Terashima, R.S.: A Modular Framework for Building Variable-Input-Length Tweakable Ciphers. In: ASIACRYPT 2013, Part I. LNCS, vol. 8269, pp. 405–423. Springer (2013)
[57] The Linux Kernel: hctr2.c. `https://github.com/torvalds/linux/blob/master/crypto/hctr2.c`, accessed: 2025-05-15
[58] Wagner, D.A.: A Generalized Birthday Problem. In: CRYPTO 2002. LNCS, vol. 2442, pp. 288–303. Springer (2002)
[59] Wang, P., Feng, D., Wu, W.: HCTR: A Variable-Input-Length Enciphering Mode. In: Information Security and Cryptology, First SKLOIS Conference, CISC 2005. LNCS, vol. 3822, pp. 175–188. Springer (2005)

Algorithm 5 Sampling Algorithm

for $\nu \in [u]$ **do** $K^{[\nu]} \xleftarrow{\$} \{0,1\}^k$ **end for**
for $\alpha \in [q]$ **do**
  $\nu \leftarrow \mathsf{u}_\alpha$; $\dot{X}_0^{(\alpha)} \leftarrow S_2^{(\alpha)} \| 01$; $\dot{X}_1^{(\alpha)} \leftarrow S_2^{(\alpha)} \| 10$; $\dot{X}_2^{(\alpha)} \leftarrow S_2^{(\alpha)} \| 11$
  $\ddot{X}_1^{(\alpha)} \leftarrow S_1^{(\alpha)} \| 000$; $\ddot{X}_2^{(\alpha)} \leftarrow S_1^{(\alpha)} \| 100$
  $\dot{Y}_0^{(\alpha)} \xleftarrow{\$} \{0,1\}^n$; $\dot{Y}_1^{(\alpha)} \xleftarrow{\$} \{0,1\}^n$; $\dot{Y}_2^{(\alpha)} \xleftarrow{\$} \{0,1\}^n$ ▷ 1st Sampling
  if $\exists \beta \in \mathcal{Q}^{[\nu]}, i, j \in [0,2]$ s.t.
    $\beta \leq \alpha, (\alpha, i) \neq (\beta, j) \wedge \dot{X}_i^{(\alpha)} \neq \dot{X}_j^{(\beta)} \wedge \dot{Y}_i^{(\alpha)} = \dot{Y}_j^{(\beta)}$ **then**
    $\dot{Y}_0^{(\alpha)} \xleftarrow{\$} \{0,1\}^n$; $\dot{Y}_1^{(\alpha)} \xleftarrow{\$} \{0,1\}^n$; $\dot{Y}_2^{(\alpha)} \xleftarrow{\$} \{0,1\}^n$ ▷ 2nd Sampling
  end if
  if $\exists \beta \in \mathcal{Q}^{[\nu]}$ s.t. $\beta < \alpha \wedge S_2^{(\alpha)} = S_2^{(\beta)}$ **then**
    $\dot{Y}_0^{(\alpha)} \leftarrow \dot{Y}_0^{(\beta)}$; $\dot{Y}_1^{(\alpha)} \leftarrow \dot{Y}_1^{(\beta)}$; $\dot{Y}_2^{(\alpha)} \leftarrow \dot{Y}_2^{(\beta)}$
  end if
  $\dot{L}^{(\alpha)} \leftarrow (\dot{Y}_0^{(\alpha)} \oplus \dot{Y}_1^{(\alpha)}) \| (\dot{Y}_0^{(\alpha)} \oplus \dot{Y}_2^{(\alpha)})$; $(\ddot{L}^{(\alpha)}, \dot{L}_+^{(\alpha)}) \xleftarrow{k} \dot{L}^{(\alpha)} \| 0^{k-n}$
  $Z_+^{(\alpha)} \xleftarrow{\$} \{0,1\}^{2n-t}$; $(Z_1^{(\alpha)}, Z_2^{(\alpha)}) \xleftarrow{n} Z^{(\alpha)} \| Z_+^{(\alpha)}$
  $\ddot{Y}_1^{(\alpha)} \leftarrow Z_1^{(\alpha)} \oplus \ddot{X}_1^{(\alpha)} \oplus \ddot{L}_+^{(\alpha)}$; $\ddot{Y}_2^{(\alpha)} \leftarrow Z_2^{(\alpha)} \oplus \ddot{X}_2^{(\alpha)}$
end for

Supplementary Materials

A Deriving the Bound of $\mathbf{Adv}^{\mathsf{dist}}_{\mathcal{O}_5,\mathcal{O}_6}(\mathsf{A})$ in Theorem 1

A.1 Coefficient-H Technique.

In this evaluation, we use the H-coefficient technique. We explain this technique.

We refer to the set of query-response tuples (and additional values) that an adversary obtains in a security game as a "transcript." Let T_5 be a transcript obtained by sampling in Game 5. Let T_6 be a transcript obtained by sampling in Game 6. We call a transcript τ *valid* if $\Pr[\mathsf{T}_6 = \tau] > 0$. Let $\mathcal{T}$ be the set of all valid transcripts such that $\forall \tau \in \mathcal{T} : \Pr[\mathsf{T}_5 = \tau] \geq \Pr[\mathsf{T}_6 = \tau]$. Then, we have $\mathsf{SD}(\mathsf{T}_5, \mathsf{T}_6) \coloneqq \sum_{\tau \in \mathcal{T}} (\Pr[\mathsf{T}_5 = \tau] - \Pr[\mathsf{T}_6 = \tau])$. By using the following lemma [53], we can derive the upper bound of the statistical distance.

Lemma 1. *Let $\mathcal{T}_{\mathsf{good}}$ and $\mathcal{T}_{\mathsf{bad}}$ be the good and bad transcripts into which $\mathcal{T}$ can be partitioned. If $\forall \tau \in \mathcal{T}_{\mathsf{good}} : \frac{\Pr[\mathsf{T}_5 = \tau]}{\Pr[\mathsf{T}_6 = \tau]} \geq 1 - \varepsilon$ s.t. $0 \leq \varepsilon \leq 1$, then $\mathsf{SD}(\mathsf{T}_5, \mathsf{T}_6) \leq \Pr[\mathsf{T}_6 \in \mathcal{T}_{\mathsf{bad}}] + \varepsilon$.*

We (1) define good and bad transcripts; (2) upper bound $\Pr[\mathsf{T}_6 \in \mathcal{T}_{\mathsf{bad}}]$; (3) lower bound $\frac{\Pr[\mathsf{T}_5 = \tau]}{\Pr[\mathsf{T}_6 = \tau]}$. Then, putting these bounds into the above lemma, we obtain the upper bound.

A.2 Sampling Algorithm of Game 6

In Game 6, we define dummy internal values by using Algorithm 5. In this algorithm, first dummy BC keys $(K^{[\nu]})_{\nu \in [u]}$ are defined. Then, for each query

index α, the plaintext blocks $\dot{X}_0^{(\alpha)}$, $\dot{X}_1^{(\alpha)}$, $\dot{X}_2^{(\alpha)}$, $\ddot{X}_1^{(\alpha)}$, $\ddot{X}_2^{(\alpha)}$ are defined according to the structure of CEtH in Lines 3 and 4. Then, dummy ciphertext blocks $(\dot{Y}_i^{(\alpha)})_{i\in[0,2]}$ of the CENC part are defined by using the resampling method. The first sampling is performed in Line 5, where the ciphertext blocks $(\dot{Y}_i^{(\alpha)})_{i\in[0,2]}$ are defined by random samplings from $\{0,1\}^n$. If a collision occurs with one of the sampled values, the second sampling is performed in Line 7, where the ciphertext blocks are resampled. If the input $S_2^{(\alpha)}$ collides with some previous one $S_2^{(\beta)}$, the β-th ciphertext blocks are assigned to the α-th ciphertext blocks in Line 11. Finally, the ciphertext blocks $(\ddot{Y}_1^{(\alpha)}, \ddot{Y}_2^{(\alpha)})$ are defined according to the structure of the Hirose part in Lines 12-14.

A.3 Definitions of Good and Bad Transcripts

We define conditions on bad transcripts in the following. The set of bad transcripts $\mathcal{T}_{\mathsf{bad}}$ is a subset of $\mathcal{T}$ such that one of the conditions holds. The set of good transcripts is defined as $\mathcal{T}_{\mathsf{good}} := \mathcal{T} \backslash \mathcal{T}_{\mathsf{bad}}$, which is the subset of $\mathcal{T}$ such that the conditions do not hold.

bad_1. The event is defined as follows.

- $\mathsf{bad}_1 \Leftrightarrow \exists \nu \in [u], \{\alpha, \beta\} \in \mathsf{UConf}_2(\mathcal{Q}^{[\nu]})$ s.t. $S_1^{(\alpha)} \| S_2^{(\alpha)} = S_1^{(\beta)} \| S_2^{(\beta)}$.

The event considers a collision of $S_1 \| S_2$ values, yielding a collision of Z values. In other words, if bad_1 does not occurs, the Z values are respectively defined by distinct samplings.

bad_2. The event is defined as follows.

- $\mathsf{bad}_2 \Leftrightarrow \exists \{(\alpha, i), (\beta, j)\} \in \mathsf{UConf}_2([q] \times [2])$ s.t.
$$(\ddot{X}_i^{(\alpha)}, \ddot{L}^{(\alpha)}) = (\ddot{X}_j^{(\beta)}, \ddot{L}^{(\beta)}) \vee (\ddot{Y}_i^{(\alpha)}, \ddot{L}^{(\alpha)}) = (\ddot{Y}_j^{(\beta)}, \ddot{L}^{(\beta)}).$$

The event considers a collision of input-output tuples of the Hirose part. Assuming that bad_2 does not occurs, one can ensure the independence of Z values defined in Game 5 as in Game 6, and avoid a collision of ciphertext blocks with the same key.

bad_3. The event is defined as follows.

- $\mathsf{bad}_3 \Leftrightarrow \exists \alpha \in [q], \beta \in [p], i \in [2]$ s.t.
$$(\ddot{L}^{(\alpha)}, \ddot{X}_i^{(\alpha)}) = (\hat{K}^{(\beta)}, \hat{X}^{(\beta)}) \vee (\ddot{L}^{(\alpha)}, \ddot{Y}_i^{(\alpha)}) = (\hat{K}^{(\beta)}, \hat{Y}^{(\beta)}).$$

The event considers a collision of IC input-output tuples between the Hirose part and offline queries. Assuming that bad_3 does not occurs, one can ensure the independence of Z values defined in Game 5 as in Game 6.

$\mathsf{fail}_{\mathsf{sample}}$. The event is defined as follows.

$$\mathsf{fail}_{\mathsf{sample}} \Leftrightarrow \exists \nu \in [u], \{(\alpha, i), (\beta, j)\} \in \mathsf{UConf}(\mathcal{Q}^{[\nu]} \times [0, 2]) \text{ s.t.} \\ \dot{X}_i^{(\alpha)} = \dot{X}_j^{(\beta)} \wedge \dot{Y}_i^{(\alpha)} = \dot{Y}_j^{(\beta)}.$$

The event considers a failure case of sampling the ciphertext blocks of the CENC part in our sampling algorithm. Hence, assuming that $\mathsf{fail}_{\mathsf{sample}}$ does not occurs, the ciphertext blocks in Game 6 are consistently defined as in Game 5.

A.4 Upper-Bounding $\Pr[\mathsf{T}_6 \in \mathcal{T}_{\mathsf{bad}}]$

Let $\mu \coloneqq \lceil n/\log_2 n \rceil$. Let $\mathbf{E} \coloneqq \{\mathsf{bad}_1, \mathsf{bad}_2, \mathsf{bad}_3, \mathsf{fail}_{\mathsf{sample}}\}$. In this evaluation, we assume that the adversary aborts just after one of $\mathbf{E}$ occurs. We then have $\Pr[\mathsf{T}_I \in \mathcal{T}_{\mathsf{bad}}] \leq \sum_{\mathsf{E} \in \mathbf{E}} \Pr[\mathsf{E}]$. The bounds of the probabilities are given as follows, providing the following bound.

$$\Pr[\mathsf{T}_6 \in \mathcal{T}_{\mathsf{bad}}] \leq \frac{3q_{\mathsf{u}}^2 q}{2^{2n}} + \frac{2n}{\log_2 n} \cdot \frac{p}{2^k} + \left(\frac{60(\log_2 n)q}{2^n}\right)^{\frac{n}{\log_2 n}} + \frac{q}{2^n} + \frac{16q_{\mathsf{u}}q}{2^{2n}} + \frac{18q^2}{2^{k+n}}.$$

Upper-Bounding $\Pr[\mathsf{bad}_1]$. For each $\nu \in [u], \{\alpha, \beta\} \in \mathsf{UConf}(\mathcal{Q}^{[\nu]})$ such that $\alpha < \beta$, either $M^{(\beta)}$ or $C^{(\beta)}$ is chosen uniformly at random from at least $2^{|M^{(\beta)}|}$ elements in $\{0,1\}^{|M^{(\beta)}|}$ such that $|M^{(\beta)}| \geq 2n$, we have $\Pr[S_1^{(\alpha)} \| S_2^{(\alpha)} = S_1^{(\beta)} \| S_2^{(\beta)}] \leq \frac{2^5}{2^{2n}}$.

Summing the bound for each possible pair (α, β), we have

$$\Pr[\mathsf{bad}_1] \leq \sum_{\nu \in [u]} \sum_{\{\alpha,\beta\} \in \mathsf{UConf}(\mathcal{Q}^{[\nu]})} \frac{2^5}{2^{2n}} \leq \frac{16q_{\mathsf{u}}q}{2^{2n}}.$$

Upper-Bounding $\Pr[\mathsf{bad}_2]$. Fix $\{(\alpha, i), (\beta, j)\} \in \mathsf{UConf}_2([q] \times [2])$ such that $\alpha \leq \beta$.

We first consider the case where $\alpha = \beta$ and $(i, j) = (1, 2)$. Since $\ddot{X}_1^{(\alpha)} \neq \ddot{X}_2^{(\alpha)}$, we have $\Pr[\ddot{X}_1^{(\alpha)} = \ddot{X}_2^{(\alpha)}] = 0$. Since $\ddot{Y}_i^{(\alpha)}$ and $\ddot{Y}_j^{(\alpha)}$ are independently chosen by random samplings from $\{0,1\}^n$. We have $\Pr[\ddot{Y}_i^{(\alpha)} = \ddot{Y}_j^{(\alpha)}] \leq \frac{1}{2^n}$.

We next consider the case where $\alpha < \beta$.

- We evaluate the collision probability $\Pr[(\ddot{X}_i^{(\alpha)}, \ddot{L}^{(\alpha)}) = (\ddot{X}_j^{(\beta)}, \ddot{L}^{(\beta)})]$. Since $\ddot{X}_1 = S_2 \| 000$ and $\ddot{X}_2 = S_2 \| 100$, we have $\Pr[(\ddot{X}_i^{(\alpha)}, \ddot{L}^{(\alpha)}) = (\ddot{X}_j^{(\beta)}, \ddot{L}^{(\beta)})] \leq \Pr[(S_1^{(\alpha)}, \ddot{L}^{(\alpha)}) = (S_1^{(\beta)}, \ddot{L}^{(\beta)})]$. By $\neg\mathsf{bad}_1$, $S_1^{(\alpha)} \neq S_1^{(\beta)}$ must hold (otherwise $S_2^{(\alpha)} \neq S_2^{(\beta)}$), ensuring that $\ddot{L}^{(\alpha)}$ and $\ddot{L}^{(\beta)}$ are independently chosen. Since either $M^{(\beta)}$ or $C^{(\beta)}$ is chosen uniformly at random from $\{0,1\}^{|M^{(\beta)}|}$, $S_2^{(\beta)}$ is chosen uniformly at random from $\{0,1\}^{k-3}$. We thus have $\Pr[(\ddot{X}_i^{(\alpha)}, \ddot{L}^{(\alpha)}) = (\ddot{X}_i^{(\beta)}, \ddot{L}^{(\beta)})] \leq \frac{2^3}{2^{k+n}}$.

– We next evaluate the collision probability $\Pr[(\ddot{Y}_i^{(\alpha)}, \ddot{L}^{(\alpha)}) = (\ddot{Y}_j^{(\beta)}, \ddot{L}^{(\beta)})]$. $\ddot{L}^{(\beta)}$ is chosen uniformly at random from $\{0,1\}^k$ and $\ddot{Y}_j^{(\beta)}$ is defined by using $Z_j^{(\beta)}$ which is chosen uniformly at random from $\{0,1\}^n$. We thus have $\Pr[(\ddot{Y}_i^{(\alpha)}, \ddot{L}^{(\alpha)}) = (\ddot{Y}_j^{(\beta)}, \ddot{L}^{(\beta)})] \leq \frac{1}{2^{n+k}}$.

Summing the above bounds for all possible tuples (α, β, i, j), we have

$$\Pr[\mathsf{bad}_2] \leq \sum_{\alpha \in [q]} \frac{1}{2^n} + \sum_{\{\alpha,\beta\} \in \mathsf{UConf}([q]), (i,j) \in [2]^2} \frac{2^3+1}{2^{k+n}} \leq \frac{q}{2^n} + \frac{18q^2}{2^{k+n}}.$$

Upper-Bounding $\Pr[\mathsf{bad}_3]$. We first define two multi-collision events. Let $\mu = \lceil n/\log_2 n \rceil$. The first event mcoll_1 (resp. mcoll_2) is for a multi-collision of plaintext (resp. ciphertext) blocks in the Hirose part. By these events, we can bound the number of keys with a specific plaintext or ciphertext block by μ.

– $\mathsf{mcoll}_1 \Leftrightarrow \exists \{\alpha_1, \ldots, \alpha_\mu\} \in \mathsf{UConf}_\mu([q])$ s.t. $S_1^{(\alpha_1)} = \cdots = S_1^{(\alpha_\mu)}$.
– $\mathsf{mcoll}_2 \Leftrightarrow \exists \{\alpha_1, \ldots, \alpha_\mu\} \in \mathsf{UConf}_\mu([q]), (i_1, \ldots, i_\mu) \in [2]^\mu$ s.t. $\ddot{Y}_{i_1}^{(\alpha_1)} = \cdots = \ddot{Y}_{i_\mu}^{(\alpha_\mu)}$.

Upper-Bounding $\Pr[\mathsf{mcoll}_1]$. Fix $\{\alpha_1, \alpha_2, \ldots, \alpha_\mu\} \in \mathsf{UConf}_\mu([q]), S_1 \in \{0,1\}^{n-3}$. For each $j \in [\mu]$, $S_1^{(\alpha_j)}$ is chosen uniformly at random from $\{0,1\}^{n-3}$, and thus we have $\Pr[S_1^{(\alpha_1)} = \cdots = S_1^{(\alpha_\mu)} = S_1] \leq \left(\frac{2^3}{2^n}\right)^\mu$. By using Stirling's approximation, we have

$$\Pr[\mathsf{mcoll}_1] \leq 2^n \binom{q}{\mu} \left(\frac{8}{2^n}\right)^\mu \leq 2^n \left(\frac{8eq}{\frac{n}{\log_2 n} \cdot 2^n}\right)^{\frac{n}{\log_2 n}} \leq \left(\frac{48(\log_2 n)q}{2^n}\right)^{\frac{n}{\log_2 n}}.$$

Upper-Bounding $\Pr[\mathsf{mcoll}_2]$. Fix $\{\alpha_1, \alpha_2, \ldots, \alpha_\mu\} \in \mathsf{UConf}_\mu([q]), i \in [2], \ddot{Y} \in \{0,1\}^n$. For each $j \in [\mu]$, $\ddot{Y}_{i_j}^{(\alpha_j)}$ is defined by using $Z_{i_j}^{(\alpha_{i_j})}$ which is chosen uniformly at random from $\{0,1\}^n$, and thus we have $\Pr[\ddot{Y}_{i_1}^{(\alpha_1)} = \ddot{Y}_{i_2}^{(\alpha_2)} = \cdots = \ddot{Y}_{i_\mu}^{(\alpha_\mu)} = \ddot{Y}] \leq \left(\frac{1}{2^n}\right)^\mu$. By using Stirling's approximation, we have

$$\begin{aligned}\Pr[\mathsf{mcoll}_2] &\leq 2^n \cdot 2^\mu \cdot \binom{q}{\mu} \left(\frac{1}{2^n}\right)^\mu \\ &\leq 2^n \left(\frac{2eq}{\frac{n}{\log_2 n} \cdot 2^n}\right)^{\frac{n}{\log_2 n}} \leq \left(\frac{12(\log_2 n)q}{2^n}\right)^{\frac{n}{\log_2 n}}.\end{aligned}$$

Deriving the Bound of $\Pr[\mathsf{bad}_3]$. We evaluate the probability that bad_3 occurs under the assumption that mcoll_1 and mcoll_2 do not occur. For each $\beta \in [p]$, the number of pairs (α, i) such that $\ddot{X}_i^{(\alpha)} = \hat{X}^{(\beta)}$ is at most $\mu - 1$, and the probability that one of the keys collides with $\hat{K}^{(\beta)}$ is at most $\frac{\mu-1}{2^k} \leq \frac{\frac{n}{\log_2 n}}{2^k}$.

Similarly, the number of pairs (α, i) such that $\ddot{Y}_i^{(\alpha)} = \hat{X}^{(\beta)}$ is at most $\mu - 1$, and the probability that one of the keys collides with $\hat{K}^{(\beta)}$ is at most $\frac{\mu-1}{2^k} \leq \frac{\frac{n}{\log_2 n}}{2^k}$. Hence, the probability that bad_3 occurs under the assumption is at most $\frac{\frac{n}{\log_2 n} p}{2^k}$.

Finally, summing the above bounds, we obtain the following bound.

$$\begin{aligned}\Pr[\mathsf{bad}_3] &\leq \frac{2n}{\log_2 n} \cdot \frac{p}{2^k} + \left(\frac{48(\log_2 n)q}{2^n}\right)^{\frac{n}{\log_2 n}} + \left(\frac{12(\log_2 n)q}{2^n}\right)^{\frac{n}{\log_2 n}} \\ &\leq \frac{2n}{\log_2 n} \cdot \frac{p}{2^k} + \left(\frac{60(\log_2 n)q}{2^n}\right)^{\frac{n}{\log_2 n}}.\end{aligned}$$

Upper-Bounding $\Pr[\mathsf{fail}_{\mathsf{sample}}]$. We first fix $\nu \in [u]$ and $\alpha \in \mathcal{Q}^{[\nu]}$, and evaluate the probability that $\mathsf{fail}_{\mathsf{sample}}$ occurs due to the ciphertext blocks of the α-th query. Since there are at most two sampling chances in the α-th query, the event for the α-th query occurs if for $s \in [2]$, in the s-th sampling of the α-th query, $\exists i_s \in [0, 2], (\beta_s, j_s) \in \mathcal{Q}^{[\nu]} \times [0, 2] \backslash \{(\alpha, i_s)\}$ such that $\beta_s < \alpha \wedge \dot{Y}_{i_s}^{(\alpha)} = \dot{Y}_{j_s}^{(\beta_s)}$. Fixing (i_1, β_1, j_1) and (i_2, β_2, j_2), since $\dot{Y}_{i_1}^{(\alpha)}$ and $\dot{Y}_{i_2}^{(\alpha)}$ are respectively chosen uniformly at random form $\{0,1\}^n$, we have $\Pr[\dot{Y}_{i_1}^{(\alpha)} = \dot{Y}_{j_1}^{(\beta_1)} \wedge \dot{Y}_{i_2}^{(\alpha)} = \dot{Y}_{j_2}^{(\beta_2)}] \leq (\frac{1}{2^n})^2$

Let $q_{<\alpha}^{[\nu]}$ be the number of queries to the ν-th before the α-th query. Summing the bound $(\frac{1}{2^n})^2$ for each of possible tuples $(\nu, \alpha, i_1, \beta_1, j_1, i_2, \beta_2, j_2)$, we have

$$\begin{aligned}\Pr[\mathsf{fail}_{\mathsf{sample}}] &\leq \sum_{\nu \in [u]} \sum_{\alpha \in \mathcal{Q}^{[\nu]}} \sum_{\substack{(\beta_1, \beta_2) \in (\mathcal{Q}^{[\nu]})^2 \\ \text{s.t. } \beta_1, \beta_2 \leq \alpha}} \left(\frac{3 \cdot 3}{2^n}\right)^2 \\ &\leq \sum_{\nu \in [u]} \sum_{\alpha \in \mathcal{Q}^{[\nu]}} \frac{9\alpha^2}{2^{2n}} \leq \sum_{\nu \in [u]} \frac{3q_\nu^3}{2^{2n}} \leq \frac{3q_{\mathsf{u}}^2 q}{2^{2n}}.\end{aligned}$$

A.5 Lower-Bounding $\frac{\Pr[\mathsf{T}_5 = \tau]}{\Pr[\mathsf{T}_6 = \tau]}$ for $\tau \in \mathcal{T}_{\mathsf{good}}$

Fix a good transcript τ. We show that $\frac{\Pr[\mathsf{T}_5 = \tau]}{\Pr[\mathsf{T}_6 = \tau]} \geq 1$. In this evaluation, we use the symbol "$\star$" for values and sets corresponding to τ. For a subset $\mathcal{S}$ of τ and $i \in \{5, 6\}$, let $\mathsf{T}_i \vdash \mathcal{S}$ be an event that T_i satisfies all elements in $\mathcal{S}$.

Evaluation of $\Pr[\mathsf{T}_i \vdash (\hat{K}^{(\alpha)\star}, \hat{X}^{(\alpha)\star}, \hat{Y}^{(\alpha)\star})_{\alpha \in [p]}]$ for $i \in \{5, 6\}$. In both games, each response to an offline query is defined by an IC, and hence $\Pr[\mathsf{T}_5 \vdash (\hat{K}^{(\alpha)\star}, \hat{X}^{(\alpha)\star}, \hat{Y}^{(\alpha)\star})_{\alpha \in [p]}] = \Pr[\mathsf{T}_6 \vdash (\hat{K}^{(\alpha)\star}, \hat{X}^{(\alpha)\star}, \hat{Y}^{(\alpha)\star})_{\alpha \in [p]}]$.

Evaluation of $\Pr[\mathsf{T}_i \vdash (K^{[\nu]\star})_{\nu \in [u]}]$ for $i \in \{5, 6\}$. In both games, each user's key is defined by the same distribution over $\{0,1\}^k$, and hence $\Pr[\mathsf{T}_5 \vdash (K^{[\nu]\star})_{\nu \in [u]}] = \Pr[\mathsf{T}_6 \vdash (K^{[\nu]\star})_{\nu \in [u]}]$.

Evaluation of $\mathbf{Pr}[\mathsf{T}_i \vdash (\ddot{Y}_j^{(\alpha)\star})_{\alpha\in[q],j\in[2]}]$ for $i \in \{5,6\}$. In Game 6, for each $\alpha \in [q]$ and $j \in [2]$, $\ddot{Y}_j^{(\alpha)}$ is chosen uniformly at random from $\{0,1\}^n$, whereas in Game 5, $\ddot{Y}_j^{\alpha}$ is defined by an E with a fresh input by $\neg(\mathsf{bad}_2 \vee \mathsf{bad}_3)$, i.e, it is freshly chosen from the space of size less than or equal to 2^n. Hence, we have $\Pr[\mathsf{T}_5 \vdash (\ddot{Y}_j^{(\alpha)\star})_{\alpha\in[q],j\in[2]}] \geq \Pr[\mathsf{T}_6 \vdash (\ddot{Y}_j^{(\alpha)\star})_{\alpha\in[q],j\in[2]}]$.

Evaluation of $\mathbf{Pr}[\mathsf{T}_i \vdash (\dot{Y}_j^{(\alpha)\star})_{\alpha\in[q],j\in[0,2]}]$ for $i \in \{5,6\}$. Fix $\alpha \in [q]$. Assume that $\mathsf{T}_i \vdash (\dot{Y}_j^{(\beta)\star})_{\beta\in[\alpha-1],j\in[0,2]}$. Let $q_{<\alpha}$ be the number of query indexes to the u_α-th user made before the α-th query.

In Game 5, $\dot{Y}_0^{(\alpha)}$, $\dot{Y}_1^{(\alpha)}$, and $\dot{Y}_2^{(\alpha)}$ are respectively chosen uniformly at random from $2^n - 3q_{<\alpha}$ elements, $2^n - (3q_{<\alpha}+1)$ elements, and $2^n - (3q_{<\alpha}+2)$ elements. Hence, we have

$$\Pr[\mathsf{T}_5 \vdash \{\dot{Y}_0^{(\alpha)}, \dot{Y}_1^{(\alpha)}, \dot{Y}_2^{(\alpha)}\}] = \frac{1}{(2^n - 3q_{<\alpha})(2^n - (3q_{<\alpha}+1))(2^n - (3q_{<\alpha}+2))}.$$

In Game 6, $\dot{Y}_0^{(\alpha)}$, $\dot{Y}_1^{(\alpha)}$, and $\dot{Y}_2^{(\alpha)}$ are respectively chosen uniformly at random from $\{0,1\}^n$ and there are at most two sampling chances by a collision with one of the other ciphertext blocks. For each sampling step, the probability that the sampled values are equal to $\dot{Y}_0^{(\alpha)\star}$, $\dot{Y}_1^{(\alpha)\star}$, and $\dot{Y}_2^{(\alpha)\star}$ is $(\frac{1}{2^n})^3$. The probability that the first sampling fails, i.e., a collision with one of the sampled values occurs is at most $\frac{3q_{<\alpha}}{2^n} + \frac{3q_{<\alpha}+1}{2^n} + \frac{3q_{<\alpha}+2}{2^n}$. Hence, we have

$$\begin{aligned}\Pr[\mathsf{T}_6 \vdash \{\dot{Y}_0^{(\alpha)\star}, \dot{Y}_1^{(\alpha)\star}, \dot{Y}_2^{(\alpha)\star}\}] &\leq \left(\frac{1}{2^n}\right)^3 + \left(\frac{1}{2^n}\right)^3 \cdot \frac{9q_{<\alpha}+3}{2^n} \\ &\leq \left(\frac{1}{2^n}\right)^3 \left(1 + \frac{9q_{<\alpha}+3}{2^n}\right).\end{aligned}$$

By the above probabilities, we have

$$\begin{aligned}&\frac{\Pr[\mathsf{T}_5 \vdash (\dot{Y}_j^{(\alpha)\star})_{j\in[0,2]}]}{\Pr[\mathsf{T}_6 \vdash (\dot{Y}_j^{(\alpha)\star})_{j\in[0,2]}]} \\ &\geq \frac{\frac{1}{(2^n-3q_{<\alpha})(2^n-(3q_{<\alpha}-1))(2^n-(3q_{<\alpha}-2))}}{\left(\frac{1}{2^n}\right)^3\left(1+\frac{9q_{<\alpha}+3}{2^n}\right)} \\ &\geq \left(\left(1-\frac{3q_{<\alpha}}{2^n}\right)\left(1-\frac{3q_{<\alpha}+1}{2^n}\right)\left(1-\frac{3q_{<\alpha}+2}{2^n}\right)\left(1+\frac{9q_{<\alpha}+3}{2^n}\right)\right)^{-1} \\ &\geq 1,\end{aligned}$$

and thus

$$\frac{\Pr[\mathsf{T}_5 \vdash (\dot{Y}_j^{(\alpha)\star})_{\alpha\in[q],j\in[0,2]}]}{\Pr[\mathsf{T}_6 \vdash (\dot{Y}_j^{(\alpha)\star})_{\alpha\in[q],j\in[0,2]}]} \geq 1.$$

Deriving the Lower-Bound of $\frac{\mathbf{Pr}[\mathsf{T}_5=\tau]}{\mathbf{Pr}[\mathsf{T}_6=\tau]}$ By the above evaluations, we have

$$\frac{\Pr[\mathsf{T}_5 = \tau]}{\Pr[\mathsf{T}_6 = \tau]} \geq 1.$$

A.6 Deriving the Bound of $\mathbf{Adv}^{\mathsf{dist}}_{\mathcal{O}_5,\mathcal{O}_6}(\mathbf{A})$

We thus have

$$\mathbf{Adv}^{\mathsf{dist}}_{\mathcal{O}_5,\mathcal{O}_6}(\mathbf{A}) \leq \frac{3q_{\mathsf{u}}^2 q}{2^{2n}} + \frac{2n}{\log_2 n} \cdot \frac{p}{2^k} + \left(\frac{60(\log_2 n)q}{2^n}\right)^{\frac{n}{\log_2 n}} + \frac{q}{2^n} + \frac{16q_{\mathsf{u}}q}{2^{2n}} + \frac{18q^2}{2^{k+n}}.$$

B Proof of Theorem 2

B.1 Definitions

We use the following definitions.

- Let $\mathcal{Q}$ be the set of all query-response tuples of $E^{\pm} = (E, E^{-1})$.
- Let $\mathsf{CENC}[\mathcal{Q}](K, S_2)$ be a function that returns an output of the CENC part with the input (K, S_2) if the corresponding plaintext and ciphertext blocks exist in $\mathcal{Q}$, i.e., if $\exists(K, S_2\|01, \dot{Y}_0)$, $(K, S_2\|10, \dot{Y}_1)$, $(K, S_2\|11, \dot{Y}_2) \in \mathcal{Q}$, then $\mathsf{CENC}[\mathcal{Q}](K, S_2) = (\dot{Y}_0 \oplus \dot{Y}_1)\|(\dot{Y}_0 \oplus \dot{Y}_2)$; otherwise $\mathsf{CENC}[\mathcal{Q}](K, S_2) = \varepsilon$.
- Let $\mathsf{Hirose}[\mathcal{Q}](\ddot{L}, S_1)$ be a function that returns an output of the Hirose part with the input $(\ddot{L}, S_1)$ if the corresponding tuples of key, plaintext, and ciphertext blocks exist in $\mathcal{Q}$, i.e., $\exists(\ddot{L}, S_1\|000, \ddot{Y}_1), (\ddot{L}, S_1\|100, \ddot{Y}_2) \in \mathcal{Q}$, then $\mathsf{Hirose}[\mathcal{Q}](\ddot{L}, S_1) = \mathsf{msb}_t((S_1\|000 \oplus \ddot{Y}_1)\|(S_1\|100 \oplus \ddot{Y}_2))$; otherwise $\mathsf{Hirose}[\mathcal{Q}](\ddot{L}, S_1) = \varepsilon$.
- Let $\mathsf{CEtH}[\mathcal{Q}](K, S_1, S_2)$ be a function that returns an output of CEtH with the input (K, S_1, S_2) if the corresponding tuples of key, plaintext, and ciphertext blocks exist in $\mathcal{Q}$; otherwise $\mathsf{CEtH}[\mathcal{Q}](K, S_1, S_2) = \varepsilon$.
- Let $\mathcal{I}_{\mathsf{CEtH}}[\mathcal{Q}]$, $\mathcal{I}_{\mathsf{CENC}}[\mathcal{Q}]$, and $\mathcal{I}_{\mathsf{Hirose}}[\mathcal{Q}]$ be respectively the set of all inputs to $\mathsf{CEtH}[\mathcal{Q}]$, $\mathsf{CENC}[\mathcal{Q}]$, and $\mathsf{Hirose}[\mathcal{Q}]$ such that the outputs are not ε.

B.2 Additional Responses

In this proof, for each query to an IC, the adversary is permitted to obtain additional input-out tuples of the IC:

- For a forward query $(\hat{K}, \hat{X})$ (resp. an inverse query $(\hat{K}, \hat{Y})$), the following input-output tuples are returned to the adversary as well as $\hat{Y} = E(\hat{K}, \hat{X})$ (resp. $\hat{X} = E^{-1}(\hat{K}, \hat{Y})$.
  - If $\mathsf{lsb}_2(\hat{X}) = 00$, then $(\hat{K}, \hat{X} \oplus 0^{n-3}100, E((\hat{K}, \hat{X} \oplus 0^{n-3}100))$.
  - If $\mathsf{lsb}_2(\hat{X}) \in \{01, 10, 11\}$, then $(\hat{K}, \mathsf{msb}_{n-2}(\hat{X})\|x, E(\hat{K}, \mathsf{msb}_{n-2}(\hat{X})\|x))_{x\in\{01,10,11\}}$.

Regarding the first response, if the query is one of input-output tuples of the Hirose part, then the other tuple is defined and returned as the additional responses. Regarding the second responses, if the query is one of input-output tuples of the CENC part, then the other tuples are defined and returned as the additional responses. Hence, the maximum number of tuples in $\mathcal{Q}$ is at most $3q$, and $|\mathcal{I}_{\mathsf{CENC}}[\mathcal{Q}]| \leq p$ and $|\mathcal{I}_{\mathsf{Hirose}}[\mathcal{Q}]| \leq p$ hold. Note that the additional responses are defined by forward queries.

B.3 Bad Events

We define several bad events for evaluating the $\mathsf{CMT}_{\mathsf{k}}$ security of CACH.

- $\mathsf{coll}_{\mathsf{CEtH}}$: This event considers a collision of $\mathsf{coll}_{\mathsf{CEtH}}$ as follows.
  - $\mathsf{coll}_{\mathsf{CEtH}} \Leftrightarrow \exists\{(K', S'_1, S'_2), (K'', S''_1, S''_2)\} \in \mathsf{UConf}_2(\mathcal{I}_{\mathsf{CEtH}}[\mathcal{Q}])$ s.t. $K' \neq K'' \wedge \mathsf{CEtH}[\mathcal{Q}](K', S'_1, S'_2) = \mathsf{CEtH}[\mathcal{Q}](K'', S''_1, S''_2)$.

Since breaking the $\mathsf{CMT}_{\mathsf{k}}$ security of CACH implies that $\mathsf{coll}_{\mathsf{CEtH}}$ occurs, we have $\mathbf{Adv}^{\mathsf{cmt_k}}_{\mathsf{CACH}}(\mathbf{A}) \leq \Pr[\mathsf{coll}_{\mathsf{CEtH}}]$.

For evaluating the probability $\Pr[\mathsf{coll}_{\mathsf{CEtH}}]$, we define the following sub-events of $\mathsf{coll}_{\mathsf{CEtH}}$.

- $\mathsf{coll}_{\mathsf{CENC}}$: This event considers a collision of the CENC part of CACH.
  - $\mathsf{coll}_{\mathsf{CENC}} \Leftrightarrow \exists\{(K', S'_2), (K'', S''_2)\} \in \mathsf{UConf}_2(\mathcal{I}_{\mathsf{CENC}}[\mathcal{Q}])$ s.t. $K' \neq K'' \wedge \mathsf{CENC}[\mathcal{Q}](K', S'_2) = \mathsf{CENC}[\mathcal{Q}](K'', S''_2)$.
- $\mathsf{coll}_{\mathsf{Hirose}}$: This event considers a collision of the Hirose part.
  - $\mathsf{coll}_{\mathsf{Hirose}} \Leftrightarrow \exists\{(\ddot{L}', S'_1), (\ddot{L}'', S''_1)\} \in \mathsf{UConf}_2(\mathcal{I}_{\mathsf{Hirose}}[\mathcal{Q}])$ s.t. $\mathsf{Hirose}[\mathcal{Q}](\ddot{L}', S'_1) = \mathsf{Hirose}[\mathcal{Q}](\ddot{L}'', S''_1)$.

B.4 Deriving the Bound of $\mathbf{Adv}^{\mathsf{cmt_k}}_{\mathsf{CACH}}(\mathbf{A})$

For $\mathsf{E}^\dagger \in \{\mathsf{coll}_{\mathsf{CENC}}, \mathsf{coll}_{\mathsf{Hirose}}, \mathsf{coll}_{\mathsf{CENC}} \vee \mathsf{coll}_{\mathsf{Hirose}} \vee (\mathsf{coll}_{\mathsf{CEtH}} \wedge \neg(\mathsf{coll}_{\mathsf{CENC}} \vee \mathsf{coll}_{\mathsf{Hirose}})\}$, let $\mathsf{E}^\dagger$ be an event that E occurs before the other events occur. The bounds of $\Pr[\mathsf{coll}^\dagger_{\mathsf{CENC}}]$, $\Pr[\mathsf{coll}^\dagger_{\mathsf{Hirose}}]$, and $\Pr[\mathsf{coll}^\dagger_{\mathsf{CEtH}}]$ are respectively given in Eqs. (2), (3), and (4). By using the bounds, we have

$$\begin{aligned}
&\mathbf{Adv}^{\mathsf{cmt_k}}_{\mathsf{CACH}}(\mathbf{A}) \leq \Pr[\mathsf{coll}^\dagger_{\mathsf{CENC}}] + \Pr[\mathsf{coll}^\dagger_{\mathsf{Hirose}}] + \Pr[\mathsf{coll}^\dagger_{\mathsf{CEtH}}] \\
&\leq \begin{cases} \dfrac{2^{2n-t} \cdot p^2}{(2^n - 3p)^2} & \text{if } k = 2n \\ 2^{2n-t} \cdot \left(\dfrac{n}{\log_2 n} \cdot \dfrac{p}{2^n - 3p}\right)^2 + \left(6(\log_2 n) \cdot \dfrac{p}{2^n - 3p}\right)^{\frac{n}{\log_2 n}} & \text{if } n \leq k < 2n. \end{cases}
\end{aligned}$$

B.5 Evaluation of $\mathbf{Pr}[\mathsf{coll}^{\dagger}_{\mathsf{CENC}}]$

Fix a pair $((K', S'_2), (K'', S''_2)) \in \mathsf{UConf}_2(\mathcal{I}_{\mathsf{CENC}}[\mathcal{Q}])$ such that $K' \neq K''$. The collision has the following structure.

$$\begin{aligned}&\mathsf{CENC}[\mathcal{Q}](K', S'_2) = \mathsf{CENC}[\mathcal{Q}](K', S'_2)\\ &\Leftrightarrow (\dot{Y}'_0 \oplus \dot{Y}'_1)\|(\dot{Y}'_0 \oplus \dot{Y}'_2) = (\dot{Y}''_0 \oplus \dot{Y}''_1)\|(\dot{Y}''_0 \oplus \dot{Y}''_2),\end{aligned}$$

where for $* \in \{\prime, \prime\prime\}$, at least two of the ciphertext blocks $\dot{Y}^*_0$, $\dot{Y}^*_1$, and $\dot{Y}^*_2$ are defined by the forward queries, which are respectively chosen uniformly at random from at least $2^n - 3p$ elements. By using the randomnesses of the ciphertext blocks, we have $\Pr[\mathsf{CENC}[\mathcal{Q}](K', S'_2) = \mathsf{CENC}[\mathcal{Q}](K', S'_2)] \leq \frac{1}{(2^n-3p)^2}$.

Summing the number of possible pairs of $\mathsf{UConf}_2(\mathcal{I}_{\mathsf{CENC}}[\mathcal{Q}])$, we have

$$\Pr[\mathsf{coll}_{\mathsf{CENC}}] \leq \binom{p}{2}\left(\frac{1}{2^n - 3p}\right)^2 \leq \frac{0.5p^2}{(2^n - 3p)^2}. \tag{2}$$

B.6 Evaluation of $\mathbf{Pr}[\mathsf{coll}^{\dagger}_{\mathsf{Hirose}}]$

Fix a pair $((\ddot{L}', S'_1), (\ddot{L}'', S''_1)) \in \mathsf{UConf}_2(\mathcal{I}_{\mathsf{Hirose}}[\mathcal{Q}])$. The collision has the following structure.

$$\begin{aligned}&\mathsf{Hirose}[\mathcal{Q}](\ddot{L}', S'_1) = \mathsf{Hirose}[\mathcal{Q}](\ddot{L}'', S'_1) \Leftrightarrow\\ &\exists W \in \{0,1\}^{2n-t} \text{ s.t. } (\ddot{X}'_1 \oplus \ddot{Y}'_1)\|(\ddot{X}'_2 \oplus \ddot{Y}'_2) = (\ddot{X}''_1 \oplus \ddot{Y}''_1)\|(\ddot{X}''_2 \oplus \ddot{Y}''_2) \oplus (0^t\|W),\end{aligned}$$

where for $* \in \{\prime, \prime\prime\}, i \in [2]$, $\ddot{X}^*_i$ or $\ddot{Y}^*_i$ is chosen uniformly at random from at least $2^n - 3p$ elements. By using the randomnesses, we have $\Pr[\mathsf{Hirose}[\mathcal{Q}](\ddot{L}', S'_1) = \mathsf{Hirose}[\mathcal{Q}](\ddot{L}'', S''_1)] \leq \frac{2^{2n-t}}{(2^n-3p)^2}$.

Summing the number of possible pairs $\mathsf{UConf}_2(\mathcal{I}_{\mathsf{Hirose}}[\mathcal{Q}])$, we have

$$\Pr[\mathsf{coll}_{\mathsf{Hirose}}] \leq 2^{2n-t}\binom{p}{2}\left(\frac{1}{2^n - 3p}\right)^2 \leq \frac{0.5p^2 2^{2n-t}}{(2^n - 3p)^2}. \tag{3}$$

B.7 Evaluation of $\mathbf{Pr}[\mathsf{coll}^{\dagger}_{\mathsf{CEtH}}]$

We evaluate the probability that a collision $\mathsf{CEtH}[\mathcal{Q}](K', S'_1, S'_2) = \mathsf{CEtH}[\mathcal{Q}](K'', S''_1, S''_2)$ occurs for some input pair $((K', S'_1, S'_2), (K'', S''_1, S''_2)) \in \mathsf{UConf}_2(\mathcal{I}_{\mathsf{CEtH}}[\mathcal{Q}])$ such that $K' \neq K''$. Note that for each $* \in \{\prime, \prime\prime\}$, we use the symbol $*$ for the values used in $\mathsf{CEtH}[\mathcal{Q}](K^*, S^*_1, S^*_2)$.

In this evaluation, we consider the following sub-events for the collision $Z' = Z''$ ($\Leftrightarrow \mathsf{CEtH}[\mathcal{Q}](K', S'_1, S'_2) = \mathsf{CEtH}[\mathcal{Q}](K'', S''_1, S''_2)$).

- $\mathsf{coll}_{\mathsf{CEtH},1} \Leftrightarrow Z' = Z'' \wedge \dot{L}' = \dot{L}''$.
- $\mathsf{coll}_{\mathsf{CEtH},2} \Leftrightarrow Z' = Z'' \wedge (\ddot{L}', S'_1) \neq (\ddot{L}'', S''_1) \wedge \ddot{L}'_+ = \ddot{L}''_+$.
- $\mathsf{coll}_{\mathsf{CEtH},3} \Leftrightarrow Z' = Z'' \wedge (\ddot{L}', S'_1) \neq (\ddot{L}'', S''_1) \wedge \ddot{L}'_+ \neq \ddot{L}''_+$.

Then, $\mathsf{coll}_{\mathsf{CEtH}} \Leftrightarrow \mathsf{coll}_{\mathsf{CEtH},1} \vee \mathsf{coll}_{\mathsf{CEtH},2} \vee \mathsf{coll}_{\mathsf{CEtH},3}$ holds, and if $\mathsf{coll}_{\mathsf{CEtH},1}$ (resp. $\mathsf{coll}_{\mathsf{CEtH},2}$) occurs, then $\mathsf{coll}_{\mathsf{CENC}}$ (resp. $\mathsf{coll}_{\mathsf{Hirose}}$) occurs. We thus have

$$\Pr[\mathsf{coll}^{\dagger}_{\mathsf{CEtH}}] \le \Pr[\mathsf{coll}_{\mathsf{CEtH},3}].$$

If $k = 2n$, then we have $\Pr[\mathsf{coll}^{\dagger}_{\mathsf{CEtH}}] = 0$. Hereafter, we assume the case that $n \le k < 2n$.

We define the following multi-collision event for $\dot{L}_1$. Let $\mu = \lceil n/\log_2 n \rceil$.

- $\mathsf{mcoll} \Leftrightarrow \exists\{(K^{[1]}, S_2^{[1]}), \ldots, (K^{[\mu]}, S_2^{[\mu]})\} \in \mathsf{UConf}_{\mu}(\mathcal{I}_{\mathsf{CENC}}[\mathcal{Q}])$ s.t. $\dot{L}_1^{[1]} = \cdots = \dot{L}_1^{[\mu]}$.

For each $i \in [\mu]$, the corresponding values used in the evaluation of $\mathsf{CENC}[\mathcal{Q}](K^{[i]}, S_2^{[i]})$ are denoted by using the superscript symbol of $[i]$. Fix $\{(K^{[1]}, S_2^{[1]}), \ldots, (K^{[\mu]}, S_2^{[\mu]})\} \in \mathsf{UConf}_{\mu}(\mathcal{I}_{\mathsf{CENC}}[\mathcal{Q}])$. For each $j \in [\mu]$, $\dot{L}_1^{[j]} = \dot{Y}_0^{[j]} \oplus \dot{Y}_1^{[j]}$ and $\dot{Y}_0^{[j]}$ or $\dot{Y}_1^{[j]}$ is chosen uniformly at random from at least $2^n - 3p$ elements by the forward query, and thus for each $W \in \{0,1\}^n$, we have $\Pr[\dot{L}_1^{[1]} = \cdots = \dot{L}_1^{[\mu]} = W] \le \left(\frac{1}{2^n - 3p}\right)^{\mu}$. By using Stirling's approximation, we have

$$\begin{aligned}\Pr[\mathsf{mcoll}] &\le 2^n \cdot \binom{p}{\mu} \left(\frac{1}{2^n - 3p}\right)^{\mu} \\ &\le 2^n \left(\frac{ep}{\frac{n}{\log_2 n} \cdot (2^n - 3p)}\right)^{\frac{n}{\log_2 n}} \le \left(\frac{6(\log_2 n)p}{2^n - 3p}\right)^{\frac{n}{\log_2 n}}.\end{aligned}$$

We then assume that mcoll does not occur.

For an input pair $((K', S_1', S_2'), (K'', S_1'', S_2'')) \in \mathsf{UConf}_2(\mathcal{I}_{\mathsf{CACH}}[\mathcal{Q}])$ such that $(\ddot{L}', S_1') \neq (\ddot{L}'', S_1'')$, since

$$Z' = Z'' \Leftrightarrow \mathsf{Hirose}[\mathcal{Q}](\ddot{L}', S_1') \oplus (\ddot{L}'_+ \| 0^{t-n}) = \mathsf{Hirose}[\mathcal{Q}](\ddot{L}'', S_1'') \oplus (\ddot{L}''_+ \| 0^{t-n})$$

by the same evaluation as $\Pr[\mathsf{coll}^{\dagger}_{\mathsf{Hirose}}]$, we have

$$\Pr[Z' = Z''] \le \frac{2^{2n-t}}{(2^n - 3p)^2}$$

By $\neg\mathsf{mcoll}$, for each input to Hirose $(\ddot{L}^*, S_1^*)$, the number of inputs to CENC $(K^\star, S_2^\star)$ such that the n-bit outputs of CENC are equal to $\ddot{L}^*$. Hence, the number of inputs to CACH that possibly occur the event $\mathsf{coll}_{\mathsf{CEtH},3}$ is at most $p(\mu - 1)$, and we have

$$\begin{aligned}\Pr[\mathsf{coll}^{\dagger}_{\mathsf{CEtH}}] &\le \binom{p(\mu-1)}{2} \cdot \frac{2^{2n-t}}{(2^n - 3p)^2} + \left(\frac{6(\log_2 n)p}{2^n - 3p}\right)^{\frac{n}{\log_2 n}} \\ &\le 2^{2n-t} \cdot \left(\frac{n}{\log_2 n} \cdot \frac{p}{2^n - 3p}\right)^2 + \left(6(\log_2 n) \cdot \frac{p}{2^n - 3p}\right)^{\frac{n}{\log_2 n}}. \qquad (4)\end{aligned}$$

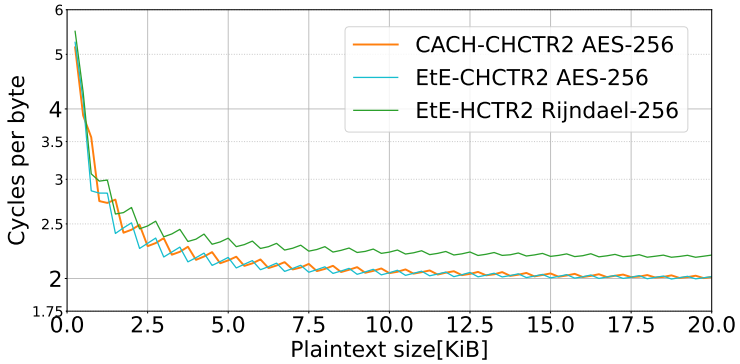

Fig. 11. Cycles per byte (cpb) of CACH-CHCTR2, EtE-CHCTR2, and EtE-HCTR2 with with different message lengths. The vertical axis: cpb on a logarithmic scale. The horizontal axis: the message size excluding the zero padding required by EtE. The periodic pattern is the artifact caused by parallel processing of multiple blocks.

C Implementation details and long-message evaluation

This appendix complements Section 8 by providing implementation details and an evaluation with larger messages.

The implementation of AES-256 is accelerated with the AES instructions [30], using 13 `vaesenc`, 1 `vaesenclast`, and 1 `vpxor` instructions. The polynomial hash function POLYVAL used within CHCTR2 is optimized using the carryless multiplication (CLMUL) instruction [31] and the aggregated reduction technique [41]. To saturate the processing pipeline, 8 and 12 message blocks are processed in parallel for AES and POLYVAL, respectively.

The Rijndael-256 implementation is also optimized using the AES and other SIMD instructions, following Drucker and Gueron [23]. The 256-bit polynomial hash function required in instantiating EtE-HCTR2 with Rijndael-256 is realized by concatenating two POLYVAL instances with different hash keys.

Fig. 11 compares the performance of the target schemes, where the vertical axis shows cycles per byte (cpb) measured with the hardware performance counters through the Linux `perf_event` API [43], and the horizontal axis shows the message size, excluding the zero padding required by EtE. The results are consistent with those in Section 8: (i) the AES-256-based schemes outperform the Rijndael-256-based one, and (ii) the cost of CACH is amortized as the message size increases, so that the performance of CACH-CHCTR2 approaches that of CHCTR2.