

Samvadini-AEAD: An Efficient, Robust, and Fully Committing AEAD Mode for Large Block Ciphers

Roberto Avanzi¹, Avik Chakraborti², Bishwajit Chakraborty³ and Eik List⁴

¹ Caesarea Rothschild Institute, University of Haifa, Israel

roberto.avanzi@icloud.com

² Institute for Advancing Intelligence, TCG Crest, Kolkata, India

avik.chakraborti@tcgcrest.org

³ School of Physical and Mathematical Sciences, Nanyang Technological University, Singapore

bishu.math.ynwa@gmail.com

⁴ Independent Researcher

elist@posteo.net

Abstract. HCTR2 is a widely used and efficient tweakable enciphering scheme, but, as shown by Chen et al. (ToSC 2023), it is difficult to design a CMT-4 secure AEAD using HCTR2 with the Encode-then-Encipher (EtE) structure. Their attack breaks CMT-4 security, and CMT-1 security remains heuristic without formal proof.

To provide these security guarantees, we first define an extension of HCTR2, named HCTR3, where the tweak is processed by a collision-resistant hash followed by a keyed AXU hash. EtE [HCTR3] can resist the aforementioned CMT-4 attack, but its CMT-4 security remains heuristic. Next, we propose a tweakable enciphering scheme, Samvadini, integrating HCTR3 with HCTR*, a simple variant of HCTR by Jha and Nandi (2022). In an EtE scheme, with the message extended with τ zeros, Samvadini defines the AEAD scheme Samvadini-AEAD. The latter is provably $n/2$ -bit robust and $\min\{n/2, \tau/2, \kappa\}$ -bit CMT-4 secure, where n , κ and τ are the block, key, and tag lengths, respectively. Its simple design surpasses the security guarantees of EtE[HCTR2], AEZ, and EtE[Adiantum], while essentially maintaining the good performance of EtE[HCTR2].

Instantiations of Samvadini with 256- or 512-bit block ciphers and efficient AXU and collision-resistant hashes meet the strictest commercial and government security standards. To facilitate such instantiations, we provide explicit examples of suitable, high-performance AXU hash functions and counter-like modes of operation based on linear feedback shift registers.

Keywords: Large Block Ciphers, Tweakable Block Ciphers, Accordion Ciphers, Authentication, Robustness, Full Commitment.

1 Introduction

Samvadini-AEAD is an efficient *robust* and *fully committing Authenticated Encryption with Associated Data* (AEAD) mode of operation. It is defined by combining Samvadini, a length-preserving *Variable Input Length* (VIL) *Tweakable Block Cipher* (TBC), with a suitable plaintext encoding. These two provably secure protocols are the main contribution of this paper.

The notion of *Robust Authenticated Encryption* (RAE) was introduced by Hoang et al. [HKR15] to allow users to choose the amount of ciphertext expansion while maintaining authenticity. RAE provides stronger security guarantees than traditional AEAD schemes,

including security against nonce misuse, tag truncation, and *Release of Unverified Plaintext* (RUP) attacks [ABL⁺14]. These security guarantees have made RAE a requisite for the recent NIST call for Accordion modes. Several RAE designs have been proposed, for instance [HKR15, CB18, CHB21, HM24, NSS24].

However, RAE does not protect against attacks such as *Partitioning Oracle Attacks* on password-based encryption [LGR21]. These attacks exploit certain encryption modes that allow efficiently finding ciphertexts that successfully decrypt under many different keys. Preventing such attacks requires *key-commitment* (CMT-1), i.e., the property that finding two plaintexts that encrypt to the same ciphertext under different keys is computationally hard. This concept was introduced by Abdalla et al. [ABN10] under the name of *robustness*.

Key-committing security is essential for use cases such as Message Franking [GLR17], Cloud Encryption, and Google Subscriptions [ADG⁺22]. Various approaches have been proposed to add commitment to existing primitives [FOR17, DGRW18, ADG⁺22]. Some applications require an even stronger property called *fully committing* security (CMT-4), i.e., it must be computationally hard to find different keys, associated data, and plaintext input tuples $(K, A, M) \neq (K', A', M')$ that encrypt to the same ciphertext [BH22]. Note that in this paper we assume that the nonce is included in the associated data. Commitment enables compact audit trails where only the tag needs to be stored. Fully committing security is also a desirable property for modes building upon accordions.

It comes as no surprise that the need for an efficient mode of operation that provides both RAE and full-committing security is widely recognized, as proven by the cited literature as well as, for instance, [BH22] and [BH24].

1.1 Robustness, Commitment, Performance: Pick only any two?

To illustrate the difficulty of combining both robustness and commitment in an efficient construction, we discuss three notable examples: AEZ [HKR15], Adiantum [CB18], and HCTR2 [CHB21].

AEZ employs a VIL TBC within the EtE (Encode-then-Encipher) framework. While it ensures RAE security, CMT-4 security is broken by a $\mathcal{O}(1)$ attack. Chen et al. [CFI⁺23] show that AEZ achieves provable *Birthday-bound* (BB) CMT-1 security only when $\tau = n$, i.e., when the tag size is full, whereas no proof exists for shorter tags. Furthermore, CMT-1 security of AEZ following the specification is practically broken in time 2^{27} .

Adiantum is a wide-block cipher primarily designed for disk and file system encryption on Android devices. It combines a universal hash and a stream cipher using a variant of the Feistel-based HSBH structure [MT05].

EtE[Adiantum], i.e., Adiantum in an EtE construction, is a RAE-secure AEAD scheme. Regarding committing security, Adiantum's guarantees vary across different versions. Chen et al. in [CFI⁺23] show that EtE[Adiantum] has $\tau/2$ bits of CMT-1 security for the zero-prepending encoding and $n/2$ bits of security for the zero-appending encoding, where τ , resp. n is the bit size of the tag, resp., block cipher state. For the zero-prepending version, Chen et al. also established a bound of $s/4$, where s is the size of the state of the stream cipher. Consequently, achieving 128-bit, resp., 256-bit, security requires a stream cipher with at a state size of at least 512, resp., 1024 bits. Finally, there exist straightforward constant-time attacks against CMT-4 security for both versions of EtE[Adiantum].

In 2021, Crowley et al. [CHB21] introduced HCTR2, a VIL TBC improving upon HCTR [WFW05]. HCTR2 combines the POLYVAL polynomial hash function with the AES block cipher and XCTR stream encryption mode. The authors proved that HCTR2 achieves BB *Tweakable Strong Pseudo-random Permutation* (TSPRP) security, and when used in the EtE construction, it provides BB RAE security. However, Chen et al. [CFI⁺23] demonstrated that EtE[HCTR2] fails to provide CMT-4 security since its *Associated Data* (AD) hash function POLYVAL lacks collision resistance in the known-key model — CMT-4 security can be broken by a simple constant-time attack. Additionally, HCTR2's use of

XCTR poses challenges for security analysis. Unlike XChaCha in Adiantum, XCTR cannot be modeled as a *Pseudo-random Function* (PRF) because it computes its counter S using the same block cipher and key as the main encryption. This allows adversaries to manipulate S to force identical block cipher inputs and create repetitions [CFI⁺23, Section 6]: This, together with the fact that the tag generation cannot be modeled as a PRF, has complicated efforts to prove EtE[HCTR2]’s BB CMT-1 security. The papers [BH22] and [HM24] tackle these gaps.

1.2 Our Solution

Observing the above-mentioned issues, we take on the problem of designing a stand-alone AEAD mode which is robust, fully committing, and efficient.

We achieve this objective in two steps:

1. First, we observe that EtE[HCTR2]’s use of an *Almost-XOR-universal* (AXU) hash function for both messages and *Associated Data* allows attacks on commitment security. We address this in EtE[HCTR3] by processing the AD with a collision-resistant cryptographic hash $\mathcal{H} : \{0, 1\}^* \rightarrow \{0, 1\}^n$, while retaining the efficient AXU hash function for plaintext and ciphertext. The collision-resistant hash function ensures AD collision resistance under known keys when combined with the AXU hash function, preventing the CMT-4 attack that works against EtE[HCTR2]. Using an AXU hash function for messages remains secure since adversaries cannot manipulate ciphertexts. Since the AD in most applications tends to be relatively short, we essentially maintain the good performance of EtE[HCTR2] — and the higher price of cryptographic hashing is otherwise a fair price to pay to attain CMT-4 security.
2. Second, we use HCTR3 to build the Samvadini-AEAD AEAD scheme, which computes the tag by taking the (unmasked) seed of the counter mode, and encrypting it. This avoids the overhead of a *Committing Concealer* [HM24].

In [BR00] it is proven that a length-preserving mode can be converted into RAE secure AEAD mode following the EtE paradigm. Our construction can be modeled as an EtE scheme using a *Hash-Encrypt-Hash* (HEH) mode, where the hash of the combination of tag and ciphertext (and AD) is computed as the sum of the hash of the tag field and hash of the ciphertext. Thus, the only difference between usual EtE encodings and ours is that instead of prepending or appending a block of zeros to the message and the tweak (or AD hash), said block is conceptually kept separate. Samvadini combines HCTR3 with a “mini”-HCTR* variant (where HCTR* [JN22, Section 7.1] substitutes XCTR with an independent PRF) which shares the lane of the first plaintext block. In the EtE scheme, this “mini”-HCTR* takes care of tag generation and HCTR3 of encryption. This allows the construction to achieve both RAE and CMT-4 security, and simplifies the CMT-1 security proof.

Samvadini provides BB security. While *Beyond-birthday Bound* (BBB) designs might seem preferable, they tend to have worse performance as well as more challenging implementation and analysis compared to BB designs using large block ciphers. Although some tweakable BBB designs like bbb-ddd-AES [DMMT24] achieve good efficiency, most tweakable BBB modes using a single n -bit block cipher per message block only provide security for up to $2^{2n/3}$ calls. With a 128-bit cipher, this corresponds to 85 bits of security. To put this value in perspective, today’s Bitcoin network performs $\approx 2^{95.2}$ SHA-256 operations annually at $750 \cdot 10^6$ TH/s [Blo24], where each “H” involves two SHA-256 calls. While gathering 2^{85} data may seem exceedingly hard, different time-memory trade-offs may be feasible, necessitating stronger security levels. On the other hand, requiring *significantly more* than 2^{128} calls to break a scheme seems to be an excessive requirement, which may needlessly impact performance.

Table 1: Comparison of our proposal and EtE-transformed existing accordion modes.

Design	RAE	Security		
		CMT-1	CMT-4	#Ops/Block
EtE[HCTR2]	$1.5 q^2 a^2 / 2^n$	BB attack	–	1 BC + 2 GF
EtE[ddd-AES]	$O(\sigma^2 / 2^{128})$	n/a	–	1 BC + 2 GF
EtE[bbb-ddd-AES]	$O(\sigma^3 / 2^{256})$	n/a	–	$(1 + \epsilon)$ BC + 2 GF
AEZ	$O(\sigma^2 / 2^n)$	$\frac{n}{2}$ (for full tag only)	–	2.5 TBC
EtE[Adiantum]	$q^2 a / 2^{116}$	BB attack	–	1 SC + 2 HF
EtE[CTET+]	$q^3 / 2^{2n}$	n/a	–	2 BC + 2 GF
EtE[DBHCTR]	$\min\left\{\frac{72 q a^2}{2^n}, \frac{6 \sigma^3 a}{2^{2n}}, \frac{12 \sigma a}{2^n}\right\}$	n/a	–	1 BC + 4 GF
EtE[PHCTR+]	$q^2 a / 2^{2n}$	n/a	–	3 TBC
EtE[ZHCTR+]	$q^2 a / 2^{2n}$	n/a	–	3 TBC
Samvadini-AEAD	$\min\{O(\sigma^2 / 2^n), 1/2^k\}$	$\min\left\{\frac{n}{2}, \frac{\tau}{2}, \kappa\right\}$	$\min\left\{\frac{n}{2}, \frac{\tau}{2}, \kappa\right\}$	1 BC + 2 GF

LEGEND: HF = Hash Function, GF = Galois Field multiplication, BC = Block Cipher, TBC = Tweakable Block Cipher, n = block size, k = key size, τ = tag length, q = number of queries, a = maximum message length in blocks, σ = total number of queried blocks over all queries, “BB attack” = No BB security proof given, a BB attack on the block size n is known, no better attacks currently known, “for full tag only” = CMT-1 security has been shown when the tag is full n -bit long, “n/a” = Not available .

Fully-secure BBB tweakable modes typically require two or more n -bit (tweakable) block cipher calls per block [Men15, BBD⁺24]. Their security bounds are often very complex expressions, with the conditions for actual BBB security level depending on several parameters. Using 256-bit wide block ciphers in simpler modes provides clearer security bounds.

Given these factors, we opted for a BB secure design using block ciphers with keys and blocks larger than 128 bits to achieve beyond 128-bit $q^2/2^n$ -type security. We recommend instantiating Samvadini with ciphers having at least 256-bit keys and blocks, such as 256-bit wide Rijndael [DR02] (possibly with a revised key schedule), Pholkos [BLLS22], a wider version of QARMAv2 [ABD⁺23, Appendix F], ASURA [TSS⁺24], or MATTER [ADM24]. Since Samvadini does not require a TBC, these ciphers, if tweakable, can be simplified by removing the tweak input and the Galois multiplications on the cells of the round keys.

1.3 Security and Efficiency Comparison with Existing Designs

We first provide a security and efficiency comparison of Samvadini-AEAD with some of the existing EtE based designs, including designs based on PHCTR+ and ZHCTR+ from [DDGN24]. The details are given in Table 1 below, where we ignore the lower order terms in the security expressions and constant overheads in the “Ops. per block” column. For example, EtE[Adiantum] has an overhead of just one block cipher invocation. We also compare Samvadini-AEAD to other important generic constructions described in [NSS24], [BH22], and [HM24]. Note that these designs, to the best of our knowledge, are more generic and are built on *Wide Block Ciphers* (WBCs) (i.e., VIL block ciphers) or CMT-1 secure designs. Precisely, [BH22] and [HM24] provide generic modes and several instances but, due to their different design paradigms, the comparison can only be generic as well, as a comparison to all possible instantiations would be too long. A more comprehensive historical perspective of VIL designs is in Appendix A.

1.3.1 Comparison with FFF [NSS24].

With FFF, Naito et al. [NSS24] take the approach of designing a robust and fully committing secure AEAD that minimizes ciphertext expansion from a WBC. More specifically, as com-

pared to a 2τ -bit ciphertext expansion needed in $\text{EtE}[\text{HCTR2}]$, $\text{EtE}[\text{AEZ}]$ and $\text{EtE}[\text{Adiantum}]$ to achieve τ -bit CMT-1 security, this approach only requires an expansion of τ bits to achieve τ -bit CMT-4 security. This is indeed a significant theoretical result. Unfortunately, as admitted by the authors, if instantiated with well-established WBCs such as AEZ, Adiantum, or HCTR2, the security is only satisfied until $\tau \leq n/2$ where n is the block size.

Thus, in applications where only CMT-1 security is required, when compared with the significant overhead added by the three variable length pseudo-random function invocations (all of which process different inputs containing the AD), the extra τ -bit ciphertext expansion seems to be a more suitable choice.

Furthermore, the problem of designing a natural construction that can achieve CMT-4 security with a negligible overhead largely remains open.

1.3.2 Comparison with the Constructions in [BH22] and [HM24].

Bellare and Hoang’s *Hash-then-Encrypt* (HtE) transform [BH22] turns a CMT-1 secure AEAD scheme into one with CMT-4 security. A *synthetic key* is created by hashing together the original key, AD (including tag length τ and nonce), and message, and then is used for message encryption. Besides requiring an underlying CMT-1-secure construction to start with, the technique needs a computationally expensive key derivation via cryptographic hashing for every query.

Later, the same authors [BH24] proposed a ciphertext-shortening transform that, given an *Authenticated Encryption* (AE) scheme that provides s bits of committing security with a ciphertext expansion of $2s$ bits, returns an AE scheme that preserves the committing security with an expansion of only s bits. However, their approach does not address RAE. Hence, the question of constructing an RAE scheme that provides s bits of committing security with a ciphertext expansion of $2s$ bits is left open.

Hoang and Menda’s EwC technique [HM24] converts a tweakable encryption scheme TE into another tweakable encryption $\overline{\text{TE}}$ such that $\text{EtE}[\overline{\text{TE}}]$ provides CMT-1 security. When $\text{EtE}[\overline{\text{TE}}]$ is used with a synthetic key, it results in a CMT-4-secure construction. Although this provides a clean modular approach to designing a fully committing secure scheme, it incurs a significant overhead over TE: it needs the derivation of four additional keys, on top of the requirements of the TE. EwC also uses a *committing concealer*, which provides CMT-4 security at the cost of a four-round Feistel using the underlying block cipher, a single invocation of a collision-resistant PRF, and two additional uses of the AXU hash function. This construction is however an important step towards finding a CMT-1 scheme, and could serve as the basis for the constructions in the two previous paragraphs.

Samvadini provides a more economical and direct approach, as its cost is essentially the same as HCTR2 with four additional single-block encryptions for value derivation and a single-block AXU hash function invocation.

1.4 Summary of Results and Outline

To efficiently process input in a HEH construction, we cannot overlook the choice of the AXU hash function. In Section 2, we generalize POLYVAL to arbitrary digest sizes — a description which seems to be missing in the literature — and highlight its connection to the Montgomery representation of binary fields. Since this hash operates entirely in the Montgomery domain, we name this generalization MHASH (Montgomery hash), in analogy to the Galois hash (GHASH). We provide examples for digest sizes of 256, 384, and 512 bits.

In Section 3, we introduce HCTR3, a tweak of HCTR2 that uses a collision-resistant cryptographic hash to reduce the AD to a single block. It exhibits tweak-collision resistance in the known-key setting, and in an EtE construction, it provides heuristic CMT-4 security.

In Section 4 we describe *future-proof* components around which to instantiate Samvadini. Besides large block ciphers and cryptographic hash functions, these also include large

AXU functions, and a variant of counter mode encryption based on *Linear Feedback Shift Registers* (LFSRs), which we call the ELK (Encrypted LFSR Keystream) mode. For both, we provide explicit instantiations.

In [Section 5](#) we introduce Samvadini-AEAD in three versions:

- (A) uses a 256-bit block cipher, 256-bit hashes, and a 256-bit master key;
- (B) uses a 512-bit block cipher, 512-bit hashes, and a 512-bit master key; and
- (C) is similar to (B) except that the master key is only 256 bits long.

In [Section 6](#) we prove that the RAE security of Samvadini-AEAD is bounded by $\min\{2^{n/2}, 2^\kappa\}$ and in [Section 7](#) that the CMT-4 security is bounded by $\min\{2^{n/2}, 2^{\tau/2}, 2^\kappa\}$. Here, n and κ denote the state size and the key size of the underlying block cipher, and τ denotes the tag length of the AEAD. Hence, Version (A) achieves robust security for up to $O(2^{128})$ queries, and Versions (B) and (C) achieve robust security for up to $O(2^{256})$ queries.

In [Section 8](#), we discuss our findings and conclude.

2 Background

2.1 Definitions and Notation

Definition 1. A block cipher $\mathbb{E} = (E, D)$ with a block size of n bits and a key size of κ bits is a pair of maps

$$\begin{aligned} E &: \mathcal{K} \times \mathcal{M} \rightarrow \mathcal{M}, (K, P) \mapsto C = E_K(P) \\ D &: \mathcal{K} \times \mathcal{M} \rightarrow \mathcal{M}, (K, C) \mapsto P = D_K(C) \end{aligned}$$

where $\mathcal{K} = \{0, 1\}^\kappa$ is the set of the *keys* and $\mathcal{M} = \{0, 1\}^n$ is the set of *blocks*, such that, for each fixed key K , the map $P \mapsto E_K(P)$ is a bijection, and $C \mapsto D_K(C)$ is the inverse of E_K . The maps E and D are called the *encryption* and *decryption* algorithms.

Furthermore, if we write this bijection as

$$P = (p_0, p_1, \dots, p_{n-1}) \mapsto C = (c_0, c_1, \dots, c_{n-1})$$

where the *input bits* p_j of P are considered as variables over $\{0, 1\}$, and the bits c_j of C viewed as functions of the p_j , no c_j can be written as a boolean function of fewer than n of the input bits, or where any input bit only appears in a linear term — in other words, the map E enjoys *full diffusion*.

In what follows, we shall always assume that, for a block cipher $\mathbb{E} = (E, D)$, both the encryption and the decryption algorithms enjoy full diffusion.

Definition 2. A *Tweakable Block Cipher* (TBC) is a pair of maps $\mathbb{E} = (E, D)$

$$\begin{aligned} E &: \mathcal{K} \times \mathcal{T} \times \mathcal{M} \rightarrow \mathcal{M}, (K, T, P) \mapsto C = E_{K,T}(P) \\ D &: \mathcal{K} \times \mathcal{T} \times \mathcal{M} \rightarrow \mathcal{M}, (K, T, C) \mapsto P = E_{K,T}(C) \end{aligned}$$

where $\mathcal{K} = \{0, 1\}^\kappa$ is the set of keys, $\mathcal{M} = \{0, 1\}^n$ is the set of messages, and $\mathcal{T} \subseteq \{0, 1\}^u$ is the set of *tweaks*, such that for any fixed tweak $T' \in \mathcal{T}$, the induced pair of maps $(E_{K,T'}, D_{K,T'})$ is a block cipher.

A *Tweakable Accordion Cipher* (TAC) is a length-preserving VIL TBC, i.e.:

Definition 3. Let $\mathcal{K}$ be a set of keys, $\mathfrak{A} \subseteq \{0, 1\}^*$ the set of *associated data* strings, and $\mathcal{M}_i = \{0, 1\}^i$ the set of the *messages* of length i . Further, let i_0 and g be two positive integers, and let $\mathcal{M} = \bigcup_{i \geq i_0, g|i} \mathcal{M}_i$ be the *message space*. A *Tweakable Accordion Cipher* (TAC) $\mathbb{E}$ is a pair of maps $\mathbb{E} = (E, D)$

$$\begin{aligned} \mathcal{E} &: \mathcal{K} \times \mathfrak{A} \times \mathcal{M}_i \rightarrow \mathcal{M}_i, \quad (K, A, P) \mapsto C = \mathcal{E}_{K,A}(P) \\ \mathcal{D} &: \mathcal{K} \times \mathfrak{A} \times \mathcal{M}_i \rightarrow \mathcal{M}_i, \quad (K, A, C) \mapsto P = \mathcal{D}_{K,A}(C) \end{aligned}$$

such that for all K, A, P, C it is $|\mathcal{E}_{K,A}(P)| = |P|$, $|\mathcal{D}_{K,A}(C)| = |C|$, and for each fixed AD $A' \in \mathfrak{A}$ and message length i' , the induced maps $\mathcal{E}_{K,A'}|_{\mathcal{M}_{i'}} : \mathcal{M}_{i'} \rightarrow \mathcal{M}_{i'}$ define a block cipher with a block size of i bits. The integer i_0 is the *minimum message length* supported by the TAC and g its *granularity*.

Note that in our definitions, the length of the tweak in a TBC is fixed, whereas the length of the AD in a TAC is allowed to vary.

Our notation is summarized in [Table 2](#). Finally, we define security levels.

Definition 4. A function is said to have a (full) k -bit security level if it can only be distinguished from a random permutation using at least $q = 2^k$ queries.

Definition 5 (Encode-then-Encipher [BR00]). Let $\mathcal{M}, \mathcal{M}^* \subseteq \{0, 1\}^*$. An *encoding scheme* on $\mathcal{M}$ (or: from $\mathcal{M}$ to $\mathcal{M}^*$) is a pair of algorithms $\text{Encode} = (\text{Encode}, \text{Decode})$ as follows.

Algorithm *Encode* can be either probabilistic, stateful, or stateless, while *Decode* is stateless and deterministic.

If *Encode* is probabilistic, whenever it is invoked on an input $M \in \mathcal{M}$ it flips some coins r , and it returns $M^* = \text{Encode}(M, r) \in \mathcal{M}^*$. It is $|\text{Encode}(M, r)| = \mathfrak{l}(|M|)$ for some function $\mathfrak{l}$, called the *length function* of the encoding scheme. If *Encode* is stateful, then r is instead derived from the state of the algorithm — for instance, a counter. The third alternative is instead that r be a fixed string, such as a sequence of zeros.

Algorithm *Decode* takes as input $M^* \in \{0, 1\}^*$ and returns either a binary string $M \in \mathcal{M}$ or the distinguished symbol $\perp$. If $M^* \notin \mathcal{M}$, then $\text{Decode}(M^*) = \perp$. If $\text{Decode}(M^*) \in \mathcal{M}$ we say that M^* is valid, otherwise we say that M^* is invalid.

For any $M \in \mathcal{M}$, it is always $\text{Decode}(\text{Encode}(M)) = M$, whereas, if $M \notin \mathcal{M}$, then $\text{Encode}(M) = \perp$.

For any TAC $\mathbb{E}$, let $\mathbb{E}^{-1}$ be the corresponding decryption algorithm. The notation $\text{EtE}[\text{Encode}, \mathbb{E}]$ denotes the composition of *Encode* with $\mathbb{E}$ together with the composition of $\mathbb{E}^{-1}$ with *Decode*. Here, $\mathbb{E}$ and $\mathbb{E}^{-1}$ are extended by defining $\mathbb{E}(\perp) = \perp$ and $\mathbb{E}^{-1}(\perp) = \perp$. If the encoding scheme is implied by the context, we simply write $\text{EtE}[\mathbb{E}]$.

We define RAE following [BMM⁺15], except that we include the nonce in the AD.

Definition 6. Let $\mathcal{K}, \mathcal{M} \subseteq \{0, 1\}^*$, $\mathcal{C} \subseteq \{0, 1\}^*$, and $\mathfrak{A} \subseteq \{0, 1\}^*$, be the sets of keys, messages, ciphertexts, and AD strings, respectively.

A *Robust Authenticated Encryption* scheme is a tuple $\mathbb{E} = (\mathcal{E}, \mathcal{D}, \lambda)$ consisting of a deterministic encryption algorithm $\mathcal{E}$, a deterministic decryption algorithm $\mathcal{D}$, and a *ciphertext expansion* $\lambda \in \mathbb{N} \cup \{0\}$

$$\begin{aligned} \mathcal{E} &: \mathcal{K} \times \mathfrak{A} \times \mathcal{M} \rightarrow \mathcal{C}, \quad (K, A, \lambda, M) \mapsto C = E_{K,A,\lambda}(M) \\ \mathcal{D} &: \mathcal{K} \times \mathfrak{A} \times \mathcal{C} \rightarrow \mathcal{M} \cup \{\perp\}, \quad (K, A, \lambda, C) \mapsto M = D_{K,A,\lambda}(M) \end{aligned}$$

such that $\mathcal{D}_{K,A,\lambda}(\mathcal{E}_{K,A,\lambda}(M)) = M$ for all K, A, λ, M and $|\mathcal{E}_{K,A,\lambda}(M)| = |M| + \lambda$.

Remark 1. In order to make the AE design *robust*, we must include the length of the tag, τ , in the AD. τ is represented as a $\nu = \lceil \log_2 n \rceil + 1$ bit string.

In [HKR15] it is stated that it is fine to set some reasonable upper bound $\lambda \leq \lambda_{\max}$, but the value $\lambda = 0$ must be allowed. In our constructions we have $\lambda \leq n$.

Definition 7 (Efficient Simulator). Let $F : \mathcal{M} \rightarrow \mathcal{M}^*$. A simulator $\mathcal{S}$ for F is *efficient* if its overhead relative to F is polynomial in the input length n , i.e., for any input x of length n , the running time of $\mathcal{S}(x)$ is bounded by $p(n) \cdot t_F(x)$, where p is a polynomial and $t_F(x)$ is the running time of F on input x .

Theorem 1 (EtE is RAE-secure [HKR15]). Let $\text{Encode} = (\text{Encode}, \text{Decode})$ be an encoding scheme satisfying $|\text{Encode}(M, a)| = |M| + a$, and $\tilde{\mathbb{E}}$ be a TAC. Then, there exist an explicitly given reduction $\mathcal{R}$ and an efficient simulator $\mathcal{S}$ with the following property: For any adversary $\mathcal{A}$, the adversary $\mathcal{B} = \mathcal{R}(\mathcal{A})$ satisfies

$$\text{Adv}_{\text{EtE}[\text{Encode}, \tilde{\mathbb{E}}], \mathcal{S}}^{\text{rae}}(\mathcal{A}) \leq \text{Adv}_{\tilde{\mathbb{E}}}^{\text{tsprp}}(\mathcal{B}) .$$

where $\mathcal{B}$ makes at most q queries for the longest tag length $\tau_{\max}$ and $\mathcal{B}$ makes at most q queries whose total length is at most that of $\mathcal{A}$'s queries plus $q \cdot \tau_{\max}$, and the running time of $\mathcal{B}$ is about that of $\mathcal{A}$ plus the encoding/decoding time.

The CMT-4 security notion immediately generalizes to robust encryption schemes by adding the tag length τ to the tuples, i.e., for a tuple (K, A, τ, M) , it must be computationally hard to find a different input tuple (K', A', τ, M') that encrypts to the same ciphertext. Now, if the tag length were allowed to differ in the two challenges, then the decryptions would have different lengths and therefore could not be equal.

We say that an AE is *tidy* [NRS14] if $M \leftarrow \mathcal{D}(K, A, \tau, C)$ implies that $\mathcal{E}(K, A, \tau, M)$ returns C . We recall the CMTD-4 security notion by Hoang et al. [HM24], where it is also proved that CMTD-4 is equivalent to CMT-4, provided that the scheme is tidy. We formulate it directly for robust schemes:

Definition 8. A robust encryption scheme $\mathbb{E} = (\mathcal{E}, \mathcal{D})$ is said to be CMTD-4 secure if there exists no efficient adversary $\mathcal{A}$ that can produce two challenge outputs of the form $(K_1, A_1, \tau, C) \neq (K_2, A_2, \tau, C)$ such that

$$\mathcal{D}(K_1, A_1, \tau, C) = M_1 \wedge \mathcal{D}(K_2, A_2, \tau, C) = M_2 .$$

Remark 2. Note that if the encryption scheme is length-preserving, then it is never committing-secure since for all ciphertext there exists a valid decryption under two different keys. Hence one can always forge an attack by computing $\mathcal{E}(K_1, A_1, 0, M_1)$ for some choice of $K_1, A_1, 0, M_1$ to receive C and then call the decryption algorithm $\mathcal{D}(K_2, A_2, 0, C)$ to generate M_2 . Therefore CMT-1 security can only be achieved if $\tau > 0$.

2.2 HCTR2

HCTR2 [CHB21], cf., Figure 1, was intended as a successor to HCTR and retains several of the features that make it an attractive design, such as simplicity and efficiency. While HCTR has a cubic security bound¹, HCTR2 achieves the quadratic security bound. HCTR2 differs from HCTR [WFW05] in four ways:

1. HCTR2 adds a value L , obtained by encrypting a fixed string, to the output of the Davies-Meyer construction on the first block, to restore the quadratic security bound;
2. HCTR2 accepts a variable-length tweak, instead of a fixed-length tweak;
3. HCTR2 replaces the hash function used in HCTR, which is not AXU [Kum18], with the POLYVAL [GLL17] AXU hash function; and
4. The hashing key is derived from the block cipher key by encrypting the all-zero block, instead of using two distinct keys for encryption and key.

¹The proof of a quadratic bound in [CN08a] contains a mistake, as discussed in [CHB21].

Table 2: Symbols and notation used in this paper.

Symbols	Meaning
A	Associated data (can be of any length).
$\mathfrak{A}$	Set of the admissible associated data strings.
Adv	Advantage.
$\mathcal{A}, \mathcal{B}$	Adversaries.
$(a)_b$	For two non-negative integers a, b , the falling factorial $a(a-1)\cdots(a-b+1)$.
bin_ℓ	Little-endian conversion of ℓ -bit integers to a binary string.
C	Ciphertext.
E, D	A fixed-size block cipher and its inverse.
$\mathcal{E}, \mathcal{D}$	Encryption and decryption algorithms.
$H, \mathcal{H}$	Hashing functions.
$\text{int}(x)$	Little-endian conversion of the binary string x to an integer.
K	(Master) key.
$\mathcal{K}$	Set of the keys.
κ	Length of the master key in bits.
ℓ	Length of a query in blocks.
λ	The ciphertext expansion.
Λ	The empty string.
M	Message.
$\mathcal{M}, \mathcal{M}^*$	Message spaces.
n	Width of the block cipher and of the hashing function.
ν	Bit length of the binary representation of τ , i.e. $\nu = \lfloor \log_2 n \rfloor + 1$.
$\mathbb{N}$	The set of natural numbers.
P	Plaintext.
σ	Bound to the total number of blocks sent in all queries.
T	Tweak.
$\mathcal{T}$	Set of the admissible tweaks.
t	Tag.
τ	Length in bits of the tag.
$ X $	Length in bits of X .
$\perp$	A special symbol returned by a function, meaning the input was invalid.
$ X _\tau$	Truncation of X to its τ least significant bits.
$\oplus$	Binary addition (XOR).
$\otimes$	Field multiplication.
$\ , \oplus$	Concatenation.
$[a, b]$	The set of natural numbers from a to b .

2.2.1 Security of HCTR2 and EtE[HCTR2].

Theorem 2 ([CHB21, Section 3.5]). *Let $\mathcal{S}$ be a simulator for HCTR2, instantiated with a block cipher E . It is*

$$\text{Adv}_{\text{HCTR2}[E]}^{\text{tsprp}}(q, \sigma, t) \leq \text{Adv}_E^{\text{sprp}}(\sigma + 2, t + \sigma t') + \frac{3\sigma^2 + 2q\sigma + q^2 + 7\sigma + 2}{2^{n+1}},$$

where n is the state size of E , t' is a small constant representing the per-block cost of simulating HCTR2, and $\sigma + 2$ bounds the number of block cipher calls made by the simulator.

Theorem 1 immediately implies that, for an encoding scheme **Encode**, there exist a reduction $\mathcal{R}$ and an efficient simulator $\mathcal{S}$ such that:

$$\text{Adv}_{\text{EtE}[\text{Encode}, \text{HCTR2}], \mathcal{S}}^{\text{rae}}(\mathcal{A}) \leq \text{Adv}_{\text{HCTR2}}^{\text{tsprp}}(\mathcal{B}).$$

Since in the CMT-1 security settings the adversary has the ability to control the key, the security only makes sense when the underlying primitives are considered as ideal. As

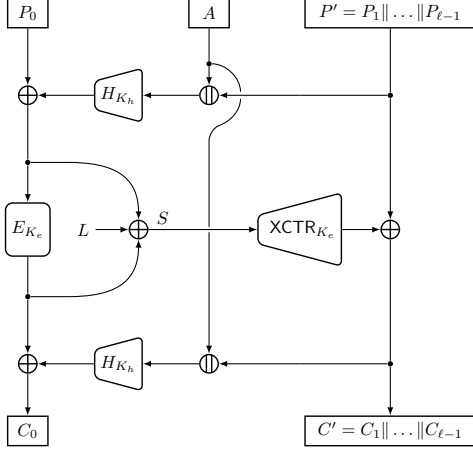


Figure 1: HCTR2 [CHB21].

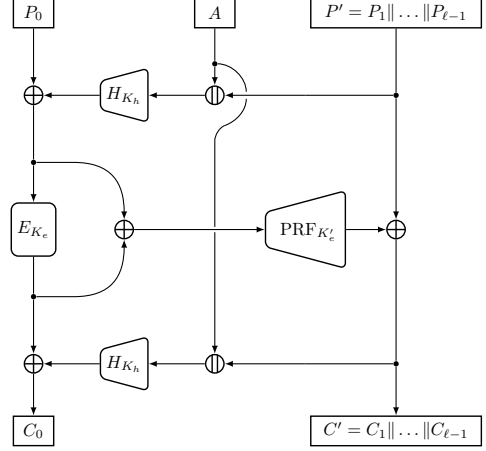


Figure 2: HCTR* [JN22].

mentioned in [CFI⁺23], the CMT-1 security of $\text{EtE}[\text{HCTR2}]$ is hard to prove when the **Encode** function is defined by prepending or appending some constant. The main reason being that the tag generated due to such an encoding is dependent on the **XCTR** function used in HCTR2, which is hard to idealize as a random function.

Furthermore, the above-mentioned instantiations of $\text{EtE}[\text{HCTR2}]$ are proven to be insecure against CMT-4 adversaries due to **POLYVAL**'s lack of preimage-resistance in the known-key setting.

2.3 HCTR*

HCTR* [JN22] is a simple variant of HCTR that achieves the quadratic security bound. Its similarities and differences w.r.t. HCTR and HCTR2 are:

1. HCTR* replaces XCTR (used in both HCTR and HCTR2) by a PRF whose key is independently sampled from the key used to encrypt the first block;
2. like HCTR, and unlike HCTR2, HCTR* does not add any L to the output of the Davies-Meyer construction on the first block;
3. like HCTR2, and unlike HCTR, HCTR* accepts a variable-length tweak; and
4. like HCTR2, and unlike HCTR, HCTR* uses an AXU [Kum18] hash function.

HCTR* is represented graphically in Figure 2.

3 HCTR3: A Natural Successor to HCTR2

We now define HCTR3, a tweak of HCTR2 designed to address **POLYVAL**'s (and **MHASH**'s) lack of preimage resistance in known-key scenarios while preserving HCTR2's *Strong Pseudo-Random Permutation* (SPRP) security.

HCTR3 is given as pseudocode in Figure 4 and graphically in Figure 3. From a master key $K \in \{0, 1\}^\kappa$ we derive two keys K_e and K_h and a mask L — in other words, there are two additional key derivations with respect to HCTR2. Furthermore, the associated data A is first reduced to a single block T using a n -bit wide collision-resistant cryptographic hash function $\mathcal{H} : \{0, 1\}^* \rightarrow \{0, 1\}^n$. The rest of the algorithm is almost the same as HCTR2, the only difference being that we use a different keystream generation method **ELK** in place of

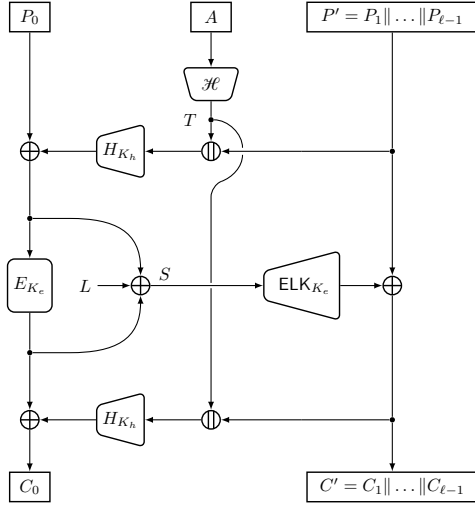


Figure 3: HCTR3.

Function: HCTR3-Encrypt**Input:** Key K , associated data A , and Plaintext P

-
- 1: $K_e \leftarrow E_K(0^n)$
 - 2: $K_h \leftarrow E_{K_e}(0^n)$
 - 3: $L \leftarrow E_{K_e}(0^{n-1} \| 1)$
 - 4: $T \leftarrow \mathcal{H}(A)$
 - 5: $P_0 \| P' \leftarrow P$ with $|P_0| = n$
 - 6: $X \leftarrow P_0 \oplus H_{K_h}(T, P')$
 - 7: $Y \leftarrow E_{K_e}(X)$
 - 8: $S \leftarrow X \oplus Y \oplus L$
 - 9: $C' \leftarrow P' \oplus \text{ELK}_{K_e}^{|P'|}(S)$
 - 10: $C_0 \leftarrow Y \oplus H_{K_h}(T, C')$
 - 11: $C \leftarrow C_0 \| C'$
 - 12: **return** C
-

Figure 4: HCTR3 encryption pseudocode (ELK is given in Figure 8).

HCTR. ELK generates the keystream by encrypting successive values of an LFSR. Efficient parameters for the implementation of this mode for $n = 256$ and $n = 512$ are provided in Subsection 4.4.

Theorem 3. *Given any SPRP Adversary $\mathcal{A}$ against HCTR3, there exists a SPRP Adversary $\mathcal{B}$ against HCTR2 and a collision Adversary $\mathcal{C}$ against $\mathcal{H}$ s.t.*

$$\text{Adv}_{\text{HCTR3}[E]_K}^{\text{sprp}}(\mathcal{A}) \leq \text{Adv}_{\text{HCTR2}[E]_K}^{\text{sprp}}(\mathcal{B}) + \text{Adv}_{\mathcal{H}}^{\text{coll}}(\mathcal{C}).$$

Proof. The proof follows from the observation that $\mathcal{H}_n$ is a cryptographic hash function. Consider the Adversaries $\mathcal{B}$, who interacts with a HCTR2 oracle, and $\mathcal{C}$, who interacts with $\mathcal{H}$ as follows:

- Whenever $\mathcal{A}$ makes an encryption (resp., decryption) query of the form (A, M) (resp., (A, C)) to $\mathcal{B}$, $\mathcal{B}$ computes $A' = \mathcal{H}(A)$ and makes an encryption (resp., decryption) query of the form (A', M) (resp., (A', C)) to its HCTR2 oracle to receive C (resp., M). It sends the same C (resp., M) to $\mathcal{A}$. Additionally, $\mathcal{C}$ also computes $A' = \mathcal{H}(A)$.
- Whenever $\mathcal{A}$ outputs the bit b , $\mathcal{B}$ outputs b . Whenever adversary $\mathcal{C}$ finds two $A_1 \neq A_2$ such that $\mathcal{H}(A_1) = \mathcal{H}(A_2)$ it outputs A_1, A_2 as its response.

To $\mathcal{A}$, $\mathcal{B}$ is a perfect simulation of a HCTR3 oracle. Now, suppose $\mathcal{C}$ does not win the collision security game against $\mathcal{H}$, i.e., for each distinct pair of AD $A_i \neq A_j$ queried by $\mathcal{A}$, it is $\mathcal{H}(A_i) \neq \mathcal{H}(A_j)$. Then, it is clear that $\mathcal{B}$ wins its SPRP game whenever $\mathcal{A}$ wins its SPRP game. $\square$

4 Future proof components

In order to build a future-proof design, we also need future-proof building blocks. In our case these are: Block ciphers with a block size of 256 bits or more; A generalisation of GHASH or, rather, of POLYVAL, to block sizes of 256 bits or more; and replacing the counter mode with a similar mode that however uses an LFSR to create the inputs to be encrypted for keystream generation. The latter is necessary because with 256- or 512-bit

blocks, not only carry propagation can become an issue in both software and hardware, and add latency comparable to a significant fraction of the block cipher’s execution time, but we want the inputs to the block ciphers to be more differentiated to hedge against cryptanalysis.

4.1 Large Block Ciphers

Several block ciphers with a block size of 256 bits or more exist, including the **Rijndael** itself — and in fact, the NIST is also considering the standardization of a 256-bit wide version of **Rijndael**. Other examples are: **Threefish** (256, 512 or 1024 bits) [FLS⁺10]; **SATURNIN** (256 bits) [CDL⁺20] and **Pholkos** (256 and 512 bits) [BLLS22], and **MATTER** [ADM24]. The tweakable ciphers can be optimized for our application by “untweaking” them, as mentioned in the Introduction, with the security analysis usually holding essentially unchanged. Such block ciphers can naturally be used to instantiate **Samvadini**.

4.2 Large Hash Functions

The CMT-4 security of **Samvadini-AEAD** relies on the collision resistance of the public hash function used. Candidates include SHA-2 [NIS15a], SHA-3 [NIS15b], BLAKE2 [ANWW13] or BLAKE3 [OANWO], and any secure prefix-free Merkle-Damgård (MD) hash [Mer79, Mer89, Dam89, CLNY06] or chopMD hash [CN08b] with a suitably large compression function — for instance, a Matyas-Meyer-Oseas compression function [MMO85] based on the same block cipher used in the mode. The latter is further motivation for the development of more efficient block ciphers.

4.3 Large AUX Hash Functions

4.3.1 Remarks on Montgomery Hashes.

POLYVAL [GLL17, CHB21] operates in the finite field $\mathbb{F}_{2^{128}}$. Similarly to **GHASH**, it processes its input by dividing it into 128-bit blocks, interpreting the resulting bit strings as field elements, and evaluates a polynomial over $\mathbb{F}_{2^{128}}$ with these field elements as coefficients and the hashing key as the evaluation point. The difference lies in the field representation: **GHASH** interprets its inputs as the coefficients of the *canonical representatives* of field elements under the isomorphism $\mathbb{F}_{2^{128}} = \mathbb{F}_2[X]/(X^{128} + X^7 + X^2 + X + 1)$. **POLYVAL** interprets all its inputs as field elements in Montgomery representation and uses Montgomery multiplication [Mon85] to reduce the cost of modular reductions.

To generalize **POLYVAL** to arbitrary digest sizes, we give a succinct treatment of Montgomery multiplication for fields of even characteristic.²

Let $p = p(X)$ be the irreducible polynomial of degree d over $\mathbb{F}_2$ used to define $\mathbb{F}_{2^d} = \mathbb{F}_2[X]/(p(X))$. In Montgomery representation, a field element with canonical representative $a \in \mathbb{F}_2[X]$ is mapped to a different polynomial that differs from a by a multiplicative factor, called the *r-sidu*:

$$\bar{a} = a \cdot X^d \bmod p .$$

The addition of two elements and testing for equality performed on r-sidus correspond to the addition and testing for equality on the original elements.

Let **REDC** be a function which, for any element $a \in \mathbb{F}_2[X]$ with $\deg(a) < 2d$, computes $a \cdot X^{-d} \bmod p$. Note that $a = \text{REDC}(\bar{a})$ for all a with $\deg(a) < d$, and that $\bar{a} = \text{REDC}(a \cdot (X^{2d} \bmod p))$, where we assume that $X^{2d} \bmod p$ is a precomputed value. Now, $\bar{a} \cdot \bar{b} \equiv a \cdot X^d \cdot b \cdot X^d \equiv a \cdot b \cdot X^d \bmod p$, whence $\bar{a} \cdot \bar{b} \cdot X^{-d} \equiv a \cdot b \bmod p$, and

$$\overline{a \cdot b} = \text{REDC}(\bar{a} \cdot \bar{b}) .$$

²Although originally described for integer arithmetic, Montgomery’s method trivially extends to finite fields, including binary fields, but this has not stopped some researchers from patenting such variants.

Function: REDC
Input: $u \in \mathbb{F}_2[X]$
Output: $u' \in \mathbb{F}_2[X]$ such that $u' \cdot X^d \equiv u \pmod{p}$

1: $\mu \leftarrow (u \bmod X^d) \cdot p' \bmod X^d$ (so $\deg \mu < d$)
2: $u' \leftarrow (u + \mu \cdot (p \bmod X^d)) / X^d$
3: **return** u'

Figure 5: Computing REDC, using a precomputed p' such that $p \cdot p' \equiv 1 \pmod{X^d}$.

Therefore, the REDC function can be used to convert elements of $\mathbb{F}_2[X]/(p(X))$ between residue representation and Montgomery domain, as well as to perform multiplications in the latter. Rather than using generic polynomial division, we compute $\text{REDC}(u)$ by adding a suitable multiple of p to make the result divisible by X^d , then perform a simple coefficient shift to divide by X^d . First, let R' and p' be monic polynomials of degrees less than d such that $X^d \cdot R' + p \cdot p' = 1$, in other words, p' is the inverse of p modulo X^d . Then, REDC is computed as described in Figure 5.

We could not find any reference for the following very useful observation.

Remark 3. If $p \bmod X^d = 1 + \sum_{i=0}^{j-1} X^{c_i}$ with $1 < d/2 \leq c_0 < c_1 < \dots < c_{j-1} < d$, then $p^2 \equiv 1 \pmod{X^d}$. Therefore, we can take $p' = p \bmod X^d = p + X^d$ and there is no need to use the extended Euclidean algorithm to compute p' .

Mathematically, POLYVAL is not a different construction from GHASH — *it is GHASH* in the Montgomery domain, up to the choice of reduction polynomial. Although the authors likely understood this connection, its absence from the original paper obscures important observations. For instance, there is no need to explicitly prove that POLYVAL is an ϵ -XOR universal hash family with $\epsilon = a/2^{128}$, where a is an upper bound on the input message length in 128-bit blocks: The result follows directly from the corresponding result for GHASH and by the fact that field multiplication by a non-zero element is a bijective linear map.

Definition 9. We use the name MHASH to denote the Montgomery multiplication-based version of GHASH: Just as the latter is a “Galois Hash”, the former is a “Montgomery Hash.” POLYVAL is the name of a specific instance of MHASH-128.

Let Λ denote the empty string. For any $d \in \mathbb{N}$ and a suitable irreducible polynomial $p(X)$ of degree d over $\mathbb{F}_2$, we define

$$\begin{cases} \text{MHASH}(h, \Lambda) = 0^d \\ \text{MHASH}(h, A \| B) = \text{REDC}(\text{POLYVAL}(h, A) + B) \cdot h \end{cases} .$$

Definition 10. While MHASH- d can be defined for any $d \in \mathbb{N}$, we fully specify instances only for $d \in \{128, 256, 384, 512\}$:

1. For $d = 128$, MHASH-128 is POLYVAL.
2. For $d = 256$, we use the primitive polynomial $p(X) = X^{256} + X^{254} + X^{251} + X^{246} + 1$. By Remark 3, it is $p' = p \bmod X^{256} = X^{254} + X^{251} + X^{246} + 1$. Using Karatsuba-Ofman multiplication [KO62], i.e.,

$$(a_1 Y + a_0)(b_1 Y + b_0) = a_1 b_1 Y^2 + ((a_1 + a_0)(b_1 + b_0) - a_0 b_0 - a_1 b_1) Y + a_0 b_0 ,$$

on top of the 128-bit polynomial multiplication, we see that processing one block of MHASH-256 costs about 3 times as much as processing one block of MHASH-128 (in

fact a bit less), and therefore it is roughly 1.5 times more expensive than MHASH-128 per bit of message. The computational cost per processed bit of MHASH-256 is 0.75 times the cost of the concatenation of two MHASH-128, i.e. POLYVAL, instances.

3. For $d = 384$ we use $p(X) = X^{384} + X^{378} + X^{369} + X^{368} + 1$. By [Remark 3](#), we have $p' = p \bmod X^{384} = X^{378} + X^{369} + X^{368} + 1$. Using the following simple short convolution (see [\[Bla85, Section 3.4\]](#) or [\[Bla10, Section 5.4\]](#))

$$\begin{aligned} (a_2Y^2 + a_1Y + a_0)(b_2Y^2 + b_1Y + b_0) \\ &= a_2b_2Y^4 + \\ &+ ((a_2 + a_1)(b_2 + b_1) - a_1b_1 - a_2b_2)Y^3 + \\ &+ ((a_2 + a_0)(b_2 + b_0) - a_0b_0 + a_1b_1 - a_2b_2)Y^2 + \\ &+ ((a_1 + a_0)(b_1 + b_0) - a_0b_0 - a_1b_1)Y + a_0b_0, \end{aligned}$$

processing one block of MHASH-384 costs about 6 times as much as processing one block of MHASH-128, and thus approximately 2 times than MHASH-128 per bit of message. The cost per bit of MHASH-384 is thus about 0.667 times the cost of three concatenated MHASH-128 instances.

4. For $d = 512$ we use $p(X) = X^{512} + X^{510} + X^{507} + X^{504} + 1$. [Remark 3](#) applies here as well, so $p' = p \bmod X^{512} = X^{510} + X^{507} + X^{504} + 1$. Using two nested Karatsuba-Ofman applications, processing one block of MHASH-512 costs about 9 times as much as processing one block of MHASH-128. Hence, MHASH-512 is 2.25 times more expensive than MHASH-128 per bit of message. MHASH-512 costs approximately 0.563 times per bit than four concatenated MHASH-128 instances.

Remark 4. In general, a multiplication of two degree $128m - 1$ binary polynomials is less expensive than m^2 multiplications of two degree 127 binary polynomials, even taking into account a few additions (which are simple carry-less XORs). Hence, a $128m$ -bit MHASH is more efficient than the concatenation of m POLYVAL instances with independent hashing keys, an approach discussed in [\[MI11, DGGP24\]](#).

Definition 11. For a hash key $h \in \{0, 1\}^d$, associated data A and message M , similarly to [\[CHB21\]](#) we define:

$$H_h(A, M) \stackrel{\text{def}}{=} \begin{cases} \text{MHASH}(h, \text{bin}(2|A| + 2) \parallel \text{pad}(A) \parallel M) & \text{if } d \text{ divides } |M| \\ \text{MHASH}(h, \text{bin}(2|A| + 3) \parallel \text{pad}(A) \parallel \text{pad}(M \parallel 1)) & \text{otherwise} \end{cases}$$

where $\text{pad}(X) := X \parallel 0^r$ and $r \equiv |X| \bmod d$ with $r \in [0, d - 1]$.

Finally, we sometimes denote $H_h(\Lambda, M)$ simply as $H_h(M)$.

4.3.2 Properties of MHASH.

Given any $(A, M) \in \{0, 1\}^* \times \{0, 1\}^*$, define $\delta(A, M) := 1 + \lceil \frac{A}{d} \rceil + \lceil \frac{M}{d} \rceil$. The following properties of $H_K(A, M)$ are proven in [\[CHB21\]](#) for GHASH, and apply to MHASH as well.

Proposition 1. For any $Z \in \{0, 1\}^d$ and any $(A, M) \in \{0, 1\}^* \times \{0, 1\}^*$,

$$\Pr_{K \xleftarrow{\$} \{0, 1\}^d} [H_K(A, M) = Z] \leq \frac{\delta(A, M)}{2^d}.$$

Proposition 2. For any $Z \in \{0, 1\}^d$ and $(A, M) \neq (A', M') \in \{0, 1\}^* \times \{0, 1\}^*$,

$$\Pr_{K \xleftarrow{\$} \{0, 1\}^d} [H_K(A, M) \oplus H_K(A', M') = Z] \leq \frac{\max\{\delta(A, M), \delta(A', M')\}}{2^d}.$$

Proposition 3. For any $Z \in \{0, 1\}^d$ and any $(A, M) \in \{0, 1\}^* \times \{0, 1\}^*$,

$$\Pr_{K \xleftarrow{\$} \{0, 1\}^d} [H_K(A, M) \oplus K = Z] \leq \frac{\delta(A, M)}{2^d} .$$

We now explore the security implications of the fact that HCTR3 processes the AD (which includes any nonce) with a cryptographic hash function.

Proposition 4. Let n be an arbitrary positive integer. Consider the MHASH hash function H used in HCTR3. Then, given any key $K_h \in \{0, 1\}^n$ and any $M \in \{0, 1\}^*$, consider an adversary $\mathcal{A}$ playing the AD-collision game against the hash function $\tilde{H}(K, A, M) := H_K(\mathcal{H}(A), M)$. Their objective is to find two $A \neq A' \in \{0, 1\}^*$ such that $\tilde{H}(K, A, M) = \tilde{H}(K, A', M)$. Then there exists a collision adversary $\mathcal{B}$ against $\mathcal{H}$ such that

$$\text{Adv}_{\tilde{H}}^{\text{AD-coll}}(\mathcal{A}) \leq \text{Adv}_{\mathcal{H}}^{\text{coll}}(\mathcal{B}) .$$

Proof. It is enough to show that, even if K, M are known, for any $A \neq A' \in \{0, 1\}^*$ we have that $\mathcal{H}(A) \neq \mathcal{H}(A')$ implies $\tilde{H}(K, A, M) \neq \tilde{H}(K, A', M)$. Note that this case is different from the analysis by Chen et al. [CFI⁺23], in the sense that for all $A \in \{0, 1\}^*$, $|\mathcal{H}(A)| = n$. Hence following the same analysis we must have given K, M known, if $\tilde{H}(K, A, M) = A$, then $\mathcal{H}(A)$ is uniquely determined as

$$\mathcal{H}(A) = \left(A \oplus 4 \otimes K^{\frac{|M|+1}{n}+1} \oplus \text{MHASH}(K, \text{pad}(M||1)) \right) \otimes K^{-\frac{|M|}{n}+1} . \quad \square$$

4.4 ELK: The Encrypted LFSR Keystream Mode of Operation

In light of cryptanalysis, the security of counter modes partially depends on how unpredictable the differences between input blocks are. If an integral attack targets a specific low-significance cell of the block cipher input, then a sufficiently long zero plaintext may enable such an attack. Similarly, *Conditional Characteristic* [BB93] may be triggered. While using full-round versions of well-designed block ciphers mitigates this, we consider the idea of using reduced-round versions for keystream generation in counter mode-based TACs, such as Samvadini. We therefore also suggest to generate the keystream by encrypting successive values of an LFSR instead of a counter. We call the resulting mode of operation *Encrypted LFSR Keystream* (ELK) mode.

All we need is to choose a suitable primitive polynomial $p(X)$ of degree n over $\mathbb{F}_2$ — for $n = 256$ and $n = 512$. We propose to use

$$\begin{aligned} &X^{256} + X^{229} + X^{193} + X^{167} + X^{139} + X^{109} + X^{71} + X^{37} + 1 , \quad \text{and} \\ &X^{512} + X^{461} + X^{397} + X^{331} + X^{281} + X^{211} + X^{139} + X^{71} + 1 . \end{aligned}$$

These two polynomials have been selected to have a good distribution of exponents. For $n = 256$ and $n = 512$, the i -th exponent is a prime number in the interval $\left[\frac{n}{8}i, \frac{n}{8}(i+1) - 1\right]$ for $i = 1, \dots, 7$. We also require that the 36 differences between exponents (including the leading term and 0) are pairwise distinct. We minimized the maximum difference between consecutive exponents to ensure each input bit has a chance to flip in the smallest possible number of iterations. This contrasts with sedimentary polynomials, where most cells simply shift one position unmodified (every eight iterations), many times in a row, potentially exposing exploitable patterns to attackers. The differences between exponents being pairwise distinct help make bit flips irregular and decorrelated between cells. These LFSRs can be efficiently implemented in both hardware and software, avoiding the carry propagation overhead of counter modes.

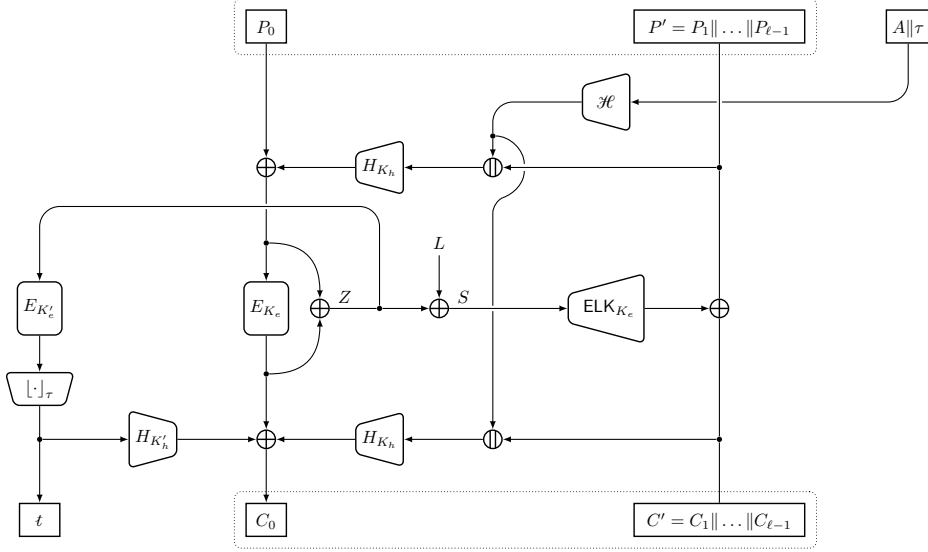


Figure 6: Samvadini-AEAD with joint ciphertext-tag length greater than $2n$.

5 Samvadini and Samvadini-AEAD

In this section, we define the TAC Samvadini, and use it to construct the AEAD scheme Samvadini-AEAD, which is both robust and fully committing.

Samvadini-AEAD builds upon the HEH paradigm established by constructions like HCTR [WFW05], FAST [CGLS22] and HCTR2 [CHB21]. These designs evolved from Naor-Reingold’s SPRP construction [NR99], replacing its internal encryption function with counter-based encryption. The addition of tweaks or AD to such constructions has been well-understood since the development of the XTR mode [MF04, MF07]. Whereas in previous constructions the tweak is incorporated by prepending it to the plaintext or the ciphertext in its entirety before processing them with an AXU hash function, in Samvadini-AEAD we use a collision-resistant cryptographic hash function to first reduce the tweak to a single block.

Samvadini-AEAD adds authentication to the HEH construction by taking a cue from *Synthetic Initialization Vector* (SIV) modes. We derive the tag by encrypting the output of the Davies-Meyer construction on the first block. We then add the hash of this encrypted value to the first ciphertext block to achieve the desired security properties.

5.1 The Samvadini Enciphering Schemes

Samvadini is parameterized by the state size n , underlying n -bit block cipher E , an AXU hash function H from $\{0, 1\}^*$ to $\{0, 1\}^n$, and a collision-resistant hash $\mathcal{H}$ from $\{0, 1\}^*$ to $\{0, 1\}^n$. Samvadini takes as input a variable length message $P \in \{0, 1\}^*$, with $|P| \geq n$, a variable length tweak $A = (A_1, \tau) \in \{0, 1\}^* \times \{0, 1\}^\nu$. Here, $\nu = \lfloor \log_2 n \rfloor + 1$ is the number of bits needed to represent all possible values of τ from 0 to n , both extremes included.

The *master key* $K \in \{0, 1\}^\kappa$ is first expanded into five values, two *encryption keys* K_e

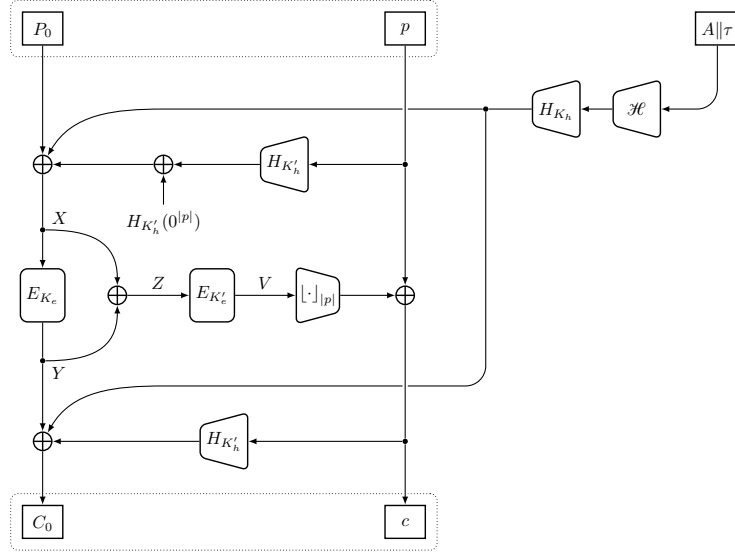


Figure 7: Samvadini-Mini.

and K'_e , two *hashing keys* K_h and K'_h , and a *mask* L , as follows:

$$\begin{cases} K_e = E_K(0^n) & (\text{or } \lfloor E_K(0^n) \rfloor_\kappa) \\ K'_e = E_K(0^{n-1} \| 1) & (\text{or } \lfloor E_K(0^{n-1} \| 1) \rfloor_\kappa) \\ K_h = E_{K_e}(0^n) \\ K'_h = E_{K'_e}(0^n) \\ L = E_{K_e}(0^{n-1} \| 1) \end{cases} \quad (1)$$

Since we are using a κ -bit master key and n -bit blocks for both block ciphers and all hashes, for $\kappa < n$ we may have the apparent contradiction of invoking the same block cipher primitive with keys of different lengths. For this reason, we provide alternative definitions of the encryption keys. If the block cipher accepts n -bit keys, there is no need to truncate K_e and K'_e to κ bits.

Depending on $|P|$, we use one of two different tweakable enciphering schemes:

- **Samvadini-Mini** is designed to handle the case of $n \leq |P| + \tau \leq 2n$. It is a close variant of HCTR*, which is defined in [JN22]. For a graphical representation of Samvadini-Mini, we refer to Figure 7.
- **Samvadini-Core** is designed to handle the case of $|P| + \tau > 2n$. It is an efficient integration of HCTR3 and (a variant of) HCTR*. For a graphical representation of Samvadini-Core, we refer to Figure 6.

Note that if we require $|P| \geq 2n$, only Samvadini-Core is used. This construction carries the advantage that *the tag computation is always completely separated from the encryption of the plaintext and the decryption of the ciphertext, and therefore the construction offers increased robustness against RUP attacks.*

As we show in Section 6, both Samvadini-Mini and Samvadini-Core enciphering schemes can be interpreted as a unified Samvadini enciphering scheme which is a combination of HCTR3 and HCTR*. With a fixed ciphertext, tag generation is independent of the XCTR encryption, which represented a stumbling block when proving CMT-1 security of existing EtE[HCTR2] schemes.

Pseudocode for Samvadini-Core and Samvadini-Mini is given in Figure 8.

Function: Samvadini-AEAD $^{n,E,H,\mathcal{H}}_K$

Input: $P, A \in \{0,1\}^*$, $|P| \geq n$, $\tau \in [0, n]$

```

1:  $P_\tau \leftarrow P \parallel 0^\tau$ 
2:  $A' \leftarrow A \parallel \tau$ 
3: if  $|P_\tau| \leq 2n$  then
4:    $C_0 \parallel c \leftarrow \mathbf{S-Mini}_K^{n,E,H,\mathcal{H}}(P_\tau, A')$ 
5:    $C_1 \parallel t \leftarrow c$  with  $|t| = \tau$ 
6:    $C \leftarrow C_0 \parallel C_1$ 
7: else
8:    $C_0 \parallel C' \parallel c \leftarrow \mathbf{S-Core}_K^{n,E,H,\mathcal{H}}(P_\tau, A')$ 
9:    $t \leftarrow c$ 
10:   $C \leftarrow C_0 \parallel C'$ 
11: return  $(C, t)$ 

```

Function: Samvadini-Core $^{n,E,H,\mathcal{H}}_K$

Input: $P \in \{0,1\}^*$ with $|P| > 2n$,
 $A \in \{0,1\}^* \times \{0,1\}^\nu$

```

1:  $K_e \leftarrow E_K(\text{bin}(0))$ 
2:  $K'_e \leftarrow E_K(\text{bin}(1))$ 
3:  $K_h \leftarrow E_{K_e}(\text{bin}(0))$ 
4:  $K'_h \leftarrow E_{K'_e}(\text{bin}(0))$ 
5:  $L \leftarrow E_{K_e}(1)$ 
6:  $\tau \leftarrow \Gamma_2(A)$ 
7: Parse  $P_0 \parallel P' \parallel p \leftarrow P$ , with  $|p| = \tau$ ,  

 $|P_0| = n$ , and  $|P'| = |P| - \tau - n$ 
8:  $T \leftarrow \mathcal{H}(A)$ 
9:  $X \leftarrow P_0 \oplus H_{K_h}(T, P') \oplus$   

 $\oplus H_{K'_h}(\Lambda, p) \oplus H_{K'_h}(\Lambda, 0^{|p|})$ 
10:  $Y \leftarrow E_{K_e}(X)$ 
11:  $Z \leftarrow X \oplus Y$ 
12:  $S \leftarrow Z \oplus L$ 
13:  $W \leftarrow \text{ELK}_{K'_e}^{|P'|}(S)$ 
14:  $V \leftarrow E_{K'_e}(Z)$ 
15:  $c \leftarrow p \oplus [V]_{|p|}$ 
16:  $C' \leftarrow P' \oplus W$ 
17:  $C_0 \leftarrow Y \oplus H_{K_h}(T, C') \oplus H_{K'_h}(\Lambda, c)$ 
18: return  $C_0 \parallel C' \parallel c$ 

```

Function: Γ_2

Input: $a \in \{0,1\}^* \times \{0,1\}^\nu$

```

1: Parse  $a' \parallel a_0 \leftarrow a$ , with  $|a_0| = \nu$ 
2: return  $\text{int}(a_0)$ 

```

Function: Samvadini-Mini $^{n,E,H,\mathcal{H}}_K$

Input: $P \in \{0,1\}^*$ with $n \leq |P| \leq 2n$,
 $A \in \{0,1\}^* \times \{0,1\}^\nu$

```

1:  $K_e \leftarrow E_K(\text{bin}(0))$ 
2:  $K'_e \leftarrow E_K(\text{bin}(1))$ 
3:  $K_h \leftarrow E_{K_e}(\text{bin}(0))$ 
4:  $K'_h \leftarrow E_{K'_e}(\text{bin}(0))$ 
5: Parse  $P_0 \parallel p \leftarrow P$ ,  

with  $|P_0| = n$ ,  $|p| = |P| - n$ 
6:  $T \leftarrow \mathcal{H}(A)$ 
7:  $X \leftarrow P_0 \oplus H_{K_h}(T, \Lambda) \oplus$   

 $\oplus H_{K'_h}(\Lambda, m) \oplus H_{K'_h}(\Lambda, 0^{|p|})$ 
8:  $Y \leftarrow E_{K_e}(X)$ 
9:  $Z \leftarrow X \oplus Y$ 
10:  $V \leftarrow E_{K'_e}(Z)$ 
11:  $c \leftarrow p \oplus [V]_{|p|}$ 
12:  $C_0 \leftarrow Y \oplus H_{K_h}(T, \Lambda) \oplus H_{K'_h}(\Lambda, c)$ 
13: return  $C_0 \parallel c$ 

```

Function: XCTR $_K^m(S)$

Input: Key K , message length m , n -bit seed S

```

1:  $U \leftarrow \Lambda$ 
2: for  $i = 0, \dots, \lfloor m/n \rfloor - 1$  do
3:    $U \leftarrow U \parallel E_K(S \oplus i)$ 
4:   if  $n \nmid m$  then
5:      $U \leftarrow U \parallel E_K(S \oplus \lfloor m/n \rfloor)[0 : (m \bmod n)])$ 
6: return  $U$ 

```

Function: ELK $_K^m(S)$

Input: Key K , message length m , n -bit seed S

```

1:  $U \leftarrow \Lambda$ 
2:  $\Sigma \leftarrow S$ 
3: for  $i = 0, \dots, \lfloor m/n \rfloor - 1$  do
4:    $U \leftarrow U \parallel E_K(\Sigma)$ 
5:   if  $i \neq \lfloor m/n \rfloor - 1$  then
6:      $\Sigma \leftarrow \text{LFSR}(\Sigma)$ 
7:   if  $n \nmid m$  then
8:      $\Sigma \leftarrow \text{LFSR}(\Sigma)$ 
9:    $U \leftarrow U \parallel E_K(\Sigma[0 : (zm \bmod n)])$ 
10: return  $U$ 

```

Figure 8: Samvadini Modes (encryption only for the sake of brevity).

5.2 The Samvadini-AEAD Mode of Operation as Encode-then-Encipher

Samvadini-AEAD can be described as an EtE design. The plaintext P , with $|P| \geq n$ is first encoded in an injective way from P to $P_\tau = P \parallel 0^\tau$. Then:

- If $|P_\tau| \leq 2n$, P_τ is processed with Samvadini-Mini, which outputs $C_0 \parallel c$. The last τ bits of $C_0 \parallel c$ form the tag t , and the first $|P_\tau| - \tau$ bits of $C_0 \parallel c$ the ciphertext C .
- If $|P_\tau| > 2n$, P_τ is processed with Samvadini-Core, which outputs $C_0 \parallel C' \parallel t$, where t is the tag with $|t| = \tau$, and $C = C_0 \parallel C'$ is the ciphertext.

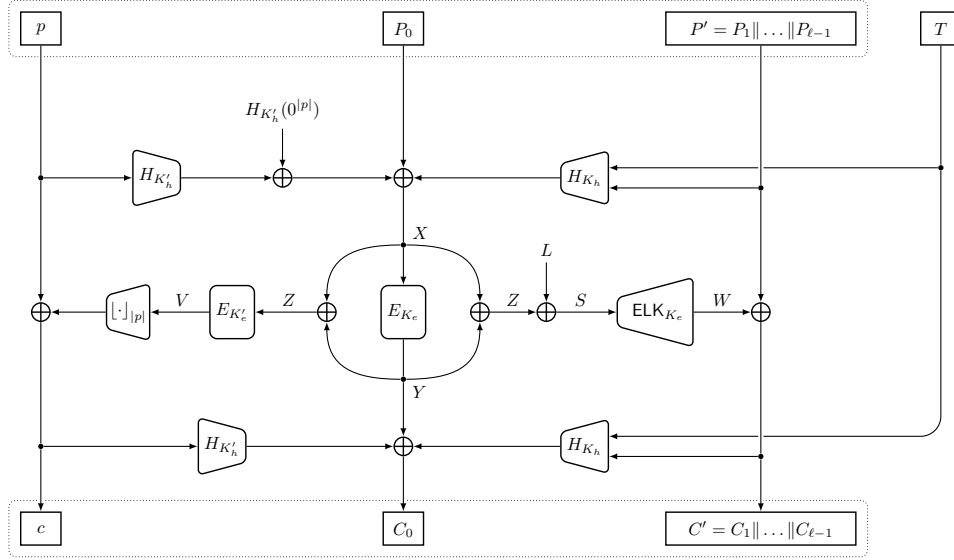


Figure 9: Samvadini Enciphering Scheme (note that $T = \mathcal{H}(A\|\tau)$).

Samvadini-AEAD is given as pseudocode in Figure 8.

5.3 Efficient Instances of Robust and CMT-4 Secure Samvadini-AEAD

In this section, we define some secure and efficient instances of Samvadini-AEAD, and discuss the possible choice of parameters n , E , H , and $\mathcal{H}$:

- **Choice of n :** We have $n \geq 256$. This is necessary, as Samvadini-AEAD achieves at most $n/2$ bits of Robust AEAD security as well as of CMT-4 security. To achieve 128 bits of security we need $n \geq 256$.
- **Choice of E :** We do not restrict the choice of E , as any n -bit *Pseudo-random Permutation* (PRP) secure encryption primitive is a good fit for Samvadini-AEAD. Examples are mentioned in Subsection 4.1.
- **Choice of $\mathcal{H}$:** The CMT-4 security of Samvadini-AEAD relies on $\mathcal{H}$ being collision resistant. Examples are given in Subsection 4.2. If the AD is just one block, it is hashed by simply using the Davies-Meyer construction around its encryption with the key K_e with the underlying block cipher. For version (C), we encrypt with the 512-bit cipher and truncate to 256 bits.
- **Choice of H :** We use the MHASH- n AXU hash function (Subsection 4.3).

We consider three choices for block and key length:

- (A) This configuration uses a 256-bit block cipher and a 256-bit master key.
The AXU hash function is MHASH-256 and $\mathcal{H}$ is also 256 bits wide.
These parameters aim at a full 128-bit security level.
- (B) The configuration uses a 512-bit block cipher and a 512-bit master key.
The AXU hash function is MHASH-512 and $\mathcal{H}$ is also 512 bits wide.
These parameters aim at a full 256-bit security level.

(C) The configuration uses a 512-bit block cipher and a 256-bit master key.

The AXU hash function is MHASH-512 and $\mathcal{H}$ is also 512 bits wide.

These parameters aim at a *practical* 256-bit security level.

Unlike schemes such as TES [Sar11], Samvadini does not require a Feistel structure on the two initial blocks since it is designed for block ciphers with large blocks (256 or 512 bits). Samvadini's decryption requires both encryption and decryption operations of the underlying block cipher. While for some cipher designs, decryption is slower than encryption, the performance impact is minimal since Samvadini's decryption needs only a single decryption operation.

6 RAE Security of Samvadini

Recall that we instantiate Samvadini-AEAD using Samvadini-Mini when the total length of ciphertext and tag is $\leq 2n$ and with Samvadini-Core otherwise. We now describe an alternative representation of the enciphering scheme that we denote by Samvadini (encompassing Samvadini-Mini and Samvadini-Core together), graphically represented in Figure 9. Samvadini-AEAD can be viewed as EtE[Encode, Samvadini] scheme. Note that, in Samvadini-AEAD, the τ zeroes are processed separately from HCTR2, and do not follow the traditional EtE techniques of appending or pretending zeroes and processing the whole encoded message, except for the short messages processed by Samvadini-Mini.

We prove the TSPRP security bound of Samvadini, which then bounds the RAE security of Samvadini-AEAD following Theorem 1.

6.1 The Samvadini-Encode function

We define the Samvadini-Encode function as follows:

$$\begin{aligned} \text{Encode} : \{0, 1\}^* \times \{0, 1\}^* \times \mathbb{N} &\rightarrow \{0, 1\}^* \times \{0, 1\}^* \\ (M, A, \tau) &\mapsto (M \parallel 0^\tau, A \parallel \text{bin}_\nu(\tau)) \end{aligned}$$

Encode is clearly injective. Given any τ , it extends the message length by τ bits and the tweak by ν bits.

6.2 Samvadini: The unified Samvadini enciphering mode

Note that, given $\tau = 0$, Samvadini-Core follows a similar mode as HCTR3 if $|P| > 2n$ and otherwise (i.e., $|P| \leq 2n$) follows a similar mode as HCTR*. Hence, in order to have a unified Samvadini, we instantiate it as a combination of HCTR* and HCTR3 conjoined at the processing of the first n -bit block.

Samvadini is a wide-block cipher which takes an n -bit block cipher E , a master-key $K \in \{0, 1\}^\kappa$, a tweak $A \in \{0, 1\}^*$, and a message $P \in \{0, 1\}^*$, with $|P| \geq n$ and proceeds as follows:

- Derive the internal keys as described in (1).
- For each encryption query:
 - (i) Parse the tweak as $A = a' \parallel a_0$, where $|a_0| = \nu$. Let τ denote the integer representation of a_0 .
 - (ii) Parse $P = P_0 \parallel P' \parallel p$ such that $|P_0| = n$ and

$$|P'| = \begin{cases} 0 & \text{if } |P| \leq 2n \\ |P| - n - \tau & \text{otherwise} \end{cases}.$$

(iii) Define

$$P'_0 := P_0 \oplus H_{K'_h}(\Lambda, p) \oplus H_{K'_h}(\Lambda, 0^{|p|}) .$$

(iv) Compute $C'_0 \| C' := \text{HCTR3}(K_e, A, P'_0 \| P')$

(v) Compute $C''_0 \| c := \text{HCTR}^*(K_e, K'_e, K'_h, P'_0 \| p)$ where HCTR^* is instantiated with the block cipher E with K_e as the encryption key, the AXU hash function H with K'_h as the hash key, and a function

$$F : \{0, 1\}^\kappa \times \{0, 1\}^n \rightarrow \{0, 1\}^{|p|} , \quad X \mapsto \lfloor E_{K'_e}(X) \rfloor_{|p|}$$

and

$$P''_0 := P_0 \oplus H_{K_h}(\mathcal{H}(A), P') \oplus H_{K'_h}(\Lambda, 0^{|p|}) .$$

(vi) Finally, the ciphertext is output as $C_0 \| C' \| c$ where

$$C_0 = C'_0 \oplus H_{K'_h}(\Lambda, c) = C''_0 \oplus H_{K_h}(\mathcal{H}(A), C') .$$

Hence, essentially, Samvadini-Enc is simply a combination of HCTR3 and HCTR*, which are joined along the encryption of the first message block. A graphical representation of Samvadini is given in [Figure 9](#).

6.3 RAE security of Samvadini-AEAD

Next, we provide the TSPRP security bound of the Samvadini enciphering scheme. The following two results show that Samvadini exhibits $\min\{n/2, \kappa\}$ -bit TSPRP security and hence Samvadini-AEAD exhibits $\min\{n/2, \kappa\}$ -bit RAE security.

Proposition 5. *The advantage $\text{Adv}_{\text{Samvadini}}^{\text{TSPRP}}(q, \sigma)$ of any TSPRP adversary making at most q queries to Samvadini consisting of up to $n\sigma$ -bit of data is at most*

$$3 \text{Adv}_E^{\text{PRP}}(\sigma + 2) + \text{Adv}_{\mathcal{H}}^{\text{coll}}(q) + \frac{4\sigma^2 + 5q\sigma + 5q^2 + 7\sigma + 2q + 1}{2^n} + \frac{1}{2^\kappa} .$$

[Proposition 5](#) is proved by adaptation of the TSPRP security proofs in [\[CHB21, JN22\]](#). Nonetheless, we provide a complete proof in [Appendix B](#).

Theorem 4. *The advantage of any RAE adversary making q queries for a total of at most $n\sigma$ bits of data to $\text{EtE}[\text{Encode}, \text{Samvadini}[E]]$ is*

$$\begin{aligned} & \text{Adv}_{\text{EtE}[\text{Encode}, \text{Samvadini}[E]]}^{\text{rae}}(q, \sigma) \\ & \leq 3 \text{Adv}_E^{\text{PRP}}(\sigma + 2) + \text{Adv}_{\mathcal{H}}^{\text{coll}}(q) + \frac{4\sigma^2 + 5q\sigma + 5q^2 + 7\sigma + 2q + 1}{2^n} + \frac{1}{2^\kappa} . \end{aligned}$$

Proof. The proof of [Theorem 4](#) follows from [Theorem 1](#) and [Proposition 5](#). $\square$

7 CMT-4/ CMTD-4 Security of Samvadini-AEAD

Here we prove the CMTD-4 security of Samvadini-AEAD, which is equivalent to the CMT-4 security.

Suppose a CMTD-4 adversary $\mathcal{A}$ outputs (K_1, A_1, τ, C) and (K_2, A_2, τ, C) . Note that for Samvadini-AEAD, since the message size is at least n bits, we must have the extended ciphertext $|C| \geq n + \tau$. Let $\ell := \lfloor \frac{|C| - \tau}{n} \rfloor$. Then, parse $C := C_0 \| \dots \| C_{\ell-1} \| t$, where $|C_k| = n$ for all $k \in [0, \ell - 1]$, $|C_\ell| = |C| - \tau - \ell \cdot n$ and $|t| = \tau$. Note that, when $n + \tau \leq |C| \leq 2n$ then $|(C_1 \| t)| \leq n$. Note that, for $i = 1, 2$, we derive the internal keys $K_{e,i}$, $K'_{e,i}$, $K_{h,i}$, and $K'_{h,i}$ from the master key K_i as in [\(1\)](#). For simplicity, let us assume that $K_{e,i}$ and $K'_{e,i}$ are κ bits long.

We consider the following adversaries:

- An adversary $\mathcal{A}_1$ playing the collision-security game against the HCTR3 hash function H as defined in [Proposition 4](#).
 - If $|C| > 2n$, then given a known key $K_{h,1}$ and a known ciphertext $(C_1 \| \dots \| C_{\ell-1})$, $\mathcal{A}_1$ generates two challenge tweak outputs, $A_1 \| \tau \neq A_2 \| \tau$.
 - If $|C| \leq 2n$, then the effective known ciphertext part of C is the empty string Λ , since $C_1 \| t$ is processed by HCTR* and no part of the ciphertext is processed by H . Then, given a known key $K_{h,1}$, $\mathcal{A}_1$ generates two challenge tweak outputs $A_1 \| \tau \neq A_2 \| \tau$.
- A restricted adversary $\mathcal{A}_2$ playing the collision-security game with the truncated (EDM) construction viewed as a compression function as follows:

$$\begin{aligned} \text{EDM}[\tau] : \{0, 1\}^\kappa \times \{0, 1\}^\kappa \times \{0, 1\}^n &\rightarrow \{0, 1\}^\tau \\ (K_e, K'_e, Y) &\mapsto \lfloor E(K'_e, Y \oplus E^{-1}(K_e, Y)) \rfloor_\tau \end{aligned}$$

where $\mathcal{A}_2$ is restricted in the sense that it is only allowed to make queries of the form (K_e, K'_e, Y) such that $K_e \neq K'_e$. Suppose $\mathcal{A}_2$ generates the challenge output pair $\{(K_{e,1}, K'_{e,1}, Y_1) \neq (K_{e,2}, K'_{e,2}, Y_2)\}$ where for each $i = 1, 2$:

$$Y_i := \begin{cases} C_0 \oplus H(K_{h,i}, \mathcal{H}(A_i \| \tau), C_1 \| \dots \| C_{\ell-1}) \oplus H(K'_h, \Lambda, t) & \text{if } |C| > 2n, \\ C_0 \oplus H(K_{h,i}, \mathcal{H}(A_i \| \tau)) \oplus H(K'_h, \Lambda, t \| C_1) & \text{otherwise.} \end{cases}$$

Define $\text{Adv}_{\text{EDM}[\tau]}^{\text{res-coll}}(\mathcal{A}_2) := \Pr[\mathcal{A}_2 \text{ wins}]$.

- An adversary $\mathcal{A}_3$ playing a bad-key collision game with E , whose objective is to produce two keys $K_1 \neq K_2$ such that $K_1 = (K_{e,1}, K'_{e,1}) = (K_{e,2}, K'_{e,2}) = K_2$. The adversary generates K_1, K_2 as its output.

Define $\text{Adv}_E^{\text{bkey-coll}}(\mathcal{A}_3) := \Pr[\mathcal{A}_3 \text{ wins}]$.

Theorem 5. *Given the adversaries $\mathcal{A}, \mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3$ as defined above,*

$$\text{Adv}_{\text{Samvadini}}^{\text{CMTD-4}}(\mathcal{A}) \leq \text{Adv}_H^{\text{AD-coll}}(\mathcal{A}_1) + \text{Adv}_{\text{EDM}[\tau]}^{\text{res-coll}}(\mathcal{A}_2) + \text{Adv}_E^{\text{bkey-coll}}(\mathcal{A}_3).$$

Proof. Suppose $\mathcal{A}$ wins the CMTD-4 game.

From the construction of Samvadini-AEAD, we must have

$$\text{EDM}[\tau](K_{e,1}, K'_{e,1}, Y_1) = \text{EDM}[\tau](K_{e,2}, K'_{e,2}, Y_2) = t.$$

- First, suppose $K_1 \neq K_2$.
 - If $(K_{e,1}, K'_{e,1}) = (K_{e,2}, K'_{e,2})$, $\mathcal{A}_3$ wins his bad-key generation game.
 - Otherwise, we have $(K_{e,1}, K'_{e,1}, Y_1) \neq (K_{e,2}, K'_{e,2}, Y_2)$. In this case the adversary $\mathcal{A}_2$ wins the restricted-collision security game against $\text{EDM}[\tau]$.
- Next, suppose $K_1 = K_2$. Then, clearly $K_{h,1} = K_{h,2}; K'_{h,1} = K'_{h,2}$. First, suppose $|C| > 2n$. Then:
 - If $H(K_{h,1}, \mathcal{H}(A_1 \| \tau), C_1 \| \dots \| C_{\ell-1}) = H(K_{h,1}, \mathcal{H}(A_2 \| \tau), C_1 \| \dots \| C_{\ell-1})$, then $\mathcal{A}_1$ wins the collision security game defined in [Proposition 4](#);
 - Otherwise, by definitions of Y_1 and Y_2 , it follows that $Y_1 \neq Y_2$. Hence the adversary $\mathcal{A}_2$ wins the res-collision security game against $\text{EDM}[\tau]$.

Now, suppose $|C| \leq 2n$. Then

- If $H(K_{h,1}, \mathcal{H}(\Lambda, A_1 \| \tau)) = H(K_{h,1}, \mathcal{H}(\Lambda, A_2 \| \tau))$, then $\mathcal{A}_1$ wins the collision security game defined in [Proposition 4](#).
- Otherwise, by the definitions of Y_1 and Y_2 , it is $Y_1 \neq Y_2$. Hence the adversary $\mathcal{A}_2$ wins the res-collision security game against $\text{EDM}[\tau]$. $\square$

Proposition 6. *Let E be an ideal block cipher. Then for any adversary making q queries to $\text{EDM}[\tau]$*

$$\text{Adv}_{\text{EDM}[\tau]}^{\text{res-coll}}(q) \leq \frac{q^2}{2^{n+1}} + \frac{q^2}{2^\tau}.$$

Proof. Consider any two queries $(K_{e,i}, K'_{e,i}, Y_i) \neq (K_{e,j}, K'_{e,j}, Y_j)$ by the adversary. Let $X_i := Y_i \oplus E_{K_{e,i}}^{-1}(Y_i)$ and $X_j := Y_j \oplus E_{K_{e,j}}^{-1}(Y_j)$. Then

$$\begin{aligned} \Pr [\text{EDM}[\tau]((K_{e,i}, K'_{e,i}, Y_i)) = \text{EDM}[\tau]((K_{e,i}, K'_{e,i}, Y_i))] \\ = \Pr \left[\lfloor E_{K'_{e,i}}(X_i) \rfloor_\tau = \lfloor E_{K'_{e,j}}(X_j) \rfloor_\tau \right]. \end{aligned}$$

- First, suppose that $K'_{e,i} \neq K'_{e,j}$. Then, in the ideal-cipher model for any values of X_i, X_j ,

$$\Pr \left[\lfloor E_{K'_{e,i}}(X_i) \rfloor_\tau = \lfloor E_{K'_{e,j}}(X_j) \rfloor_\tau \right] \leq \frac{1}{2^\tau}.$$

- Next, suppose $K'_{e,i} = K'_{e,j}$. Then,

$$\begin{aligned} \Pr \left[E_{K'_{e,i}}(X_i) = E_{K'_{e,j}}(X_j) \right] \\ = \Pr[X_i = X_j] + \Pr \left[\lfloor E_{K'_{e,i}}(X_i) \rfloor_\tau = \lfloor E_{K'_{e,j}}(X_j) \rfloor_\tau \mid X_i \neq X_j \right] \\ = \Pr \left[E_{K_{e,i}}(X_i \oplus Y_i) = E_{K_{e,j}}(X_j \oplus Y_j) \mid X_i = X_j \right] + \frac{2}{2^\tau}. \end{aligned}$$

- First suppose $K_{e,i} = K_{e,j}$. Then, it holds that $Y_i \neq Y_j$ and hence

$$\Pr \left[E_{K_{e,i}}(X_i \oplus Y_i) = E_{K_{e,i}}(X_j \oplus Y_j) \mid X_i = X_j \right] = 0.$$

- Next, suppose $K_{e,i} \neq K_{e,j}$. Then, in the ideal-cipher model:

$$\Pr \left[E_{K_{e,i}}(X_i \oplus Y_i) = E_{K_{e,j}}(X_j \oplus Y_j) \mid X_i = X_j \right] \leq \frac{1}{2^n}.$$

Hence, varying over all $i \neq j$ we obtain the result. $\square$

Proposition 7. *Let E be an ideal block cipher. Then for any adversary making q queries to E*

$$\text{Adv}_E^{\text{bkey-coll}}(q) \leq \frac{q^2}{2^{2\kappa}}.$$

Proof. The proof follows from the observation that given any two key queries $K_i \neq K_j$, the adversary wins if and only if $\lfloor E_{K_1}(\text{bin}(0)) \rfloor_\kappa = \lfloor E_{K_2}(\text{bin}(0)) \rfloor_\kappa$ and $\lfloor E_{K_1}(\text{bin}(1)) \rfloor_\kappa = \lfloor E_{K_2}(\text{bin}(1)) \rfloor_\kappa$. Now, in the ideal-cipher model, given any $Y_0 \neq Y_1$ and $K \in \{0, 1\}^\kappa$

$$\Pr \left[(\lfloor E_K(\text{bin}(0)) \rfloor_\kappa = Y_0) \wedge (\lfloor E_K(\text{bin}(1)) \rfloor_\kappa = Y_1) \right] \leq \frac{1}{2^\kappa} \times \frac{1}{2^\kappa - 1} \leq \frac{2}{2^{2\kappa}}.$$

Hence,

$$\begin{aligned} \text{Adv}_E^{\text{bkey-coll}}(q) &\leq \sum_{j_1 \neq j_2 \in [1, q]} \sum_{(Y_0 \neq Y_1) \in \{0, 1\}^\kappa} \prod_{i=1, 2} \Pr \left[\begin{aligned} &(\lfloor E_{K_{j_i}}(\text{bin}(0)) \rfloor_\kappa = Y_0) \wedge \\ &\wedge (\lfloor E_{K_{j_i}}(\text{bin}(1)) \rfloor_\kappa = Y_1) \end{aligned} \right] \\ &\leq \frac{q^2}{2} \times \frac{2^{2\kappa}}{2} \times \frac{4}{2^{4\kappa}} = \frac{q^2}{2^{2\kappa}}. \end{aligned} \quad \square$$

7.1 Final Bound on $\text{Adv}_{\text{Samvadini-AEAD}}^{\text{CMT-4}}(\mathcal{A})$

Finally, following [Theorem 5](#), [Proposition 4](#), [Proposition 6](#) and [Proposition 7](#), we get that, in the ideal cipher model,

$$\text{Adv}_{\text{Samvadini-AEAD}}^{\text{CMT-4}}(q) \leq \text{Adv}_{\mathcal{H}}^{\text{coll}}(q) + \frac{q^2}{2^{n+1}} + \frac{q^2}{2^\tau} + \frac{q^2}{2^{2\kappa}}.$$

8 Conclusion, Discussion, and Open Questions

We have introduced Samvadini-AEAD, a robust and fully committing AEAD mode, which is conceptually simple and based on well-established principles. We intentionally designed it to achieve a BB security bound. Instantiated with a 256 or 512-bit block cipher, it can provide security against up to 2^{128} or 2^{256} queries.

We observe that BBB security could perhaps be achieved in a similar framework, for instance by adopting the XORP construction as in Iwata’s CENC mode [\[Iwa06, IMV16\]](#). This consists in adding a mask to each block of the keystream. Like the keystream blocks, the mask is the encryption of a value of the LFSR, but is only refreshed every 2^j blocks. This would give $n - j$ bits of PRF security. However, it would likely require other changes in the design to maintain robustness and full commitment, and it is not obvious how to achieve this and maintain efficiency. We leave this as an open question for future work.

Compared to [EtE\[HCTR2\]](#), we require just three extra block cipher calls while achieving stronger security. The impact of using a cryptographic hash for AD is limited since the AD is typically small. Our overhead is also low compared to the use of transforms like [EwC \[HM24\]](#).

To our knowledge, at the time of this writing, Samvadini-AEAD is the first explicitly instantiated robust and fully committing AEAD mode. Recent work on 256- and 512-bit block ciphers makes practical deployment of Samvadini possible. Furthermore, we provide instances of wide AXU hash functions ([MHASH](#)) and LFSRs for efficient keystream generation. While we have not recommended a cryptographic hash to be used to process the AD, we note that a prefix-free MD hash using a Matyas-Meyer-Oseas compression function based on the mode’s underlying block cipher is an interesting option as it reduces the dependencies required by software implementations.

For BB designs using 256-bit or 512-bit block ciphers to outperform BBB modes using AES-256, the large block ciphers must be faster than two and four AES-256 calls, respectively. While easy in hardware, matching AES performance in software remains challenging on platforms with AES instructions. Notably, even [Rijndael-256-256](#) is roughly three to five times slower than AES-256 [\[DG22\]](#). If we compare to *full-security* BBB modes, the thresholds reach four and eight AES-256 calls, respectively, and obtaining better performance with Samvadini may already be possible — but CENC-like modes reduce those thresholds again at the price of just a few bits of security. While we are not aware of robust and CMT-4 BBB modes of operation in the literature, the challenges we just mentioned highlight the need for more efficient large block cipher designs.

Using round-reduced versions of the block cipher in the keystream generation can significantly accelerate the mode. Since adversaries cannot control the inputs to keystream generation, many attacks do not apply, such as Yoyo attacks [\[RBH17\]](#), which can be otherwise devastating on round-reduced large block ciphers. For example, [Pholkos-256/512 \[BLLS22\]](#) could use only 10/12 rounds for keystream generation, and the full 16/20 rounds elsewhere. The feasibility of this approach depends on cryptanalysis, and while the cryptanalysis of [Pholkos \[BLLS22\]](#) seems to support it, more work is needed. This may enable Samvadini to outperform full-security BBB AES-256-based modes while providing a stronger security model.

Table 3: Survey of Variable Input Length Encryption Constructions.

Design	Security	Type of hash	Operation count	Ref.
NR	$q^2 a^2 / (2^n - 1)$	Invertible blockwise AXU	m BC + 2 HF	[NR99]
CMC	$7 q^2 a^2 / 2^n$	—	$(2m + 1)$ BC	[HR03]
EME	$7 q^2 a^2 / 2^n$	—	$(2m + 2)$ BC	[HR04]
ABL3	$O(q^2 a^2 / 2^n)$	Polynomial	m BC + 2 HF	[MV04]
ABL4	$O(q^2 a^2 / 2^n)$	Polynomial	$2m$ BC + 2 HF	[MV04]
XCB	Broken [BVA24, WMX+24]	Polynomial	$(m + 1)$ BC + 2 HF	[MF04]
OCB1	$9.5 q^2 a^2 / 2^n + 2^{n-\tau} / (2^n - 1)$	—	$(m + 2)$ BC	[Rog04]
HCTR	$q^3 a^2 / 2^n$	Polynomial	m BC + 2 HF	[WFW05]
HCH	$7 q^2 a^2 / 2^n$	Polynomial	$(m + 3)$ BC + 2 HF	[CS06a]
PEP	$(q^2 + 6 q^2 (a + 5)^2) / 2^{n+1}$	Polynomial	$(m + 5)$ BC + $(4m - 6)$ GF	[CS06b]
TET	$q^2 a^2 / (2^n - 1)$	Invertible blockwise AXU	$(m + 1)$ BC + $2m$ GF	[Hal07]
HSE	$5 q^2 a^2 / 2^n$	AXU	m BC + 2 HF	[MM07]
HMCH	$v q^2 a^2 / 2^n$	Polynomial ($v = 4$) or BRW ($v = 10$), AXU	$(m + 2)$ BC + 2 HF	[Sar09]
LBC1	$3 q^2 a^2 / 2^{2n}$	Polynomial, AXU	$(m - 1)$ TBC + 4 HF	[MI11]
OCB3	$6 q^2 a^2 / 2^n + 2^{n-\tau} / (2^n - 1)$	—	$(m + 2)$ BC	[KR11]
TCT ₂	$q a^2 / 2^n + 6 a q^3 / (2^{2n-2} - a^3 q^3) +$ $+ 1296 q^3 / (2^{2n-2} - 216 q^3)$	Polynomial, AU/AXU	$(6m + 6)$ BC + $+ (7m + 7)$ HF	[ST13]
ZMAC	$2.5 q^2 a^2 / 2^{2n + \min\{s, n\}} + 4 q / 2^n$	—	$(6 + m)$ TBC	[IMPS17]
ZCZ	$3 q^2 a^2 / 2^{2n+1}$	Polynomial	$(2m + 6)$ TBC + 4 HF	[BLN18]
Adiantum	$q^2 a / 2^{116}$	Polynomial, Δ U	BC + SC + 2 HF	[CB18]
THCTR	$O((qa)^2 / 2^n)$ as per [ABPV21]	n -bit keyed AXU and $(n + s^*)$ -bit keyed pAXU	m TBC + 3 HF	[DN18]
Deck	$q^2 / 2^{127}$	XU	2 VIL-SC + 2 HF	[GDM19]
HCTR2	$1.5 q^2 a^2 / 2^n$	Polynomial, AXU	m BC + 2 HF	[CHB21]
CTET+	$O(q^3 / 2^{2n})$	—	$2m$ BC + $2m$ GF	[CEL+21]
FAST	$7 q^2 a^2 / 2^n$	BRW	m BC + 2 HF	[CGLS22]
TIE-plus	$q^2 a^2 / 2^n \chi + q^2 a^2 (1 - 1/\chi) / 2^n +$ $+ 2 q^2 a / 2^n$	Masking function	$2m$ MF + m PERM	[Zha22]
XOCB	$am / 2^n + am^3 / 2^{2n}$	—	$(m + 8)$ BC	[BHI+23] [Li�23]
ACCOR	$q^2 a^2 / 2^{129}$	Polynomial, AXU	m BC + 2 HF	[DFUB24]
ACCOR-L	$qa\chi / 2^{128}$	Polynomial, AXU	m BC + 4 GF + 2 HF	[DFUB24]
DbHCTR	$\min\{72 q a^2 / 2^n, 6 \sigma^3 a / 2^{2n}, 12 \sigma a / 2^n\}$	Polynomial, AXU	m BC + $4m$ GF + 2 HF	[Lee24]
ddd-AES	$q^2 a / 2^n$	Polynomial, AXU	$(m + 2)$ BC + 2 HF	[DMMT24]
bbb-ddd-AES	$O(q^3 / 2^{2n})$	Polynomial, AXU	$(2m + 4)$ BC + 2 HF	[DMMT24]

LEGEND: HF = Hash Function, BC = Block Cipher, TBC = Tweakable Block Cipher, SC = Stream Cipher, VIL-SC = Variable Input Length SC, PERM = Permutation, GF = Galois Field multiplication, MF = Masking Function, XU = XOR Universal, AXU = Almost XOR Universal, AU = Almost Universal, Δ U = Almost Δ Universal, pAXU = partial AXU, n = block size, τ = tag length, χ = number of keys, q = number of queries, s^* = tweak length after hashing, a = maximum message length in blocks, m = number of blocks of a specific message, σ = total number of queried blocks over all queries, BRW = Bernstein–Rabin–Winograd Polynomials [RW72, Ber07].

Appendices/Supporting Material

A Brief Overview of Variable Input Length Encryption

In Table 3 we summarize the principal VIL encryption constructions with particular focus on the length-preserving designs, as well as some that provide integrity as well. This table has been taken from [DFUB24], updated and expanded.

B Proof of Proposition 5

In this section, we provide the TSPRP security bound of the Samvadini enciphering scheme. Throughout this section, for Samvadini, we use the alphabets A , P , and C to denote the tweak, the message, and the ciphertext. For any $i \in [1, q]$ and a variable X , we use X^i to denote the value of the variable during the i^{th} query.

Let G_0 represent the original Samvadini construction. Let G_1 be the modified G_0 construction where K_e and K'_e are sampled uniformly randomly from $\{0, 1\}^\kappa$. Let G_2 be the modified G_1 construction where $E_{K'_e}$ is replaced by a random permutation π_1 . Let G_3 be the modified G_2 construction where all the E_{K_e} (including the E_{K_e} calls used in the ELK_{K_e}) are replaced by another random permutation π_2 that is independent from π_1 .

Lemma 1. *The following bound holds:*

$$\text{Adv}_{\text{Samvadini}[E]}^{\text{tsprp}}(q, \sigma) \leq 3 \cdot \text{Adv}_E^{\text{prp}}(\sigma + 2) + \text{Adv}_{G_3}^{\text{tsprp}}(q, \sigma) + \frac{1}{2^\kappa} .$$

Proof. We prove the lemma by the game-hopping technique. From the triangle inequality

$$\begin{aligned} \left| \text{Adv}_{\text{Samvadini}[E]}^{\text{tsprp}}(q, \sigma) - \text{Adv}_{G_3}^{\text{tsprp}}(q, \sigma) \right| &\leq \\ &\leq \left| \text{Adv}_{\text{Samvadini}[E]}^{\text{tsprp}}(q, \sigma) - \text{Adv}_{G_1}^{\text{tsprp}}(q, \sigma) \right| \\ &\quad + \left| \text{Adv}_{G_1}^{\text{tsprp}}(q, \sigma) - \text{Adv}_{G_2}^{\text{tsprp}}(q, \sigma) \right| + \left| \text{Adv}_{G_2}^{\text{tsprp}}(q, \sigma) - \text{Adv}_{G_3}^{\text{tsprp}}(q, \sigma) \right| . \end{aligned}$$

The only difference between G_0 and G_1 is that in G_0 , K_e and K'_e are the outputs of $E_K(\text{bin}(0))$ and $E_K(\text{bin}(1))$ whereas in G_1 , they are generated uniformly at random, i.e., the two calls to E_K are replaced with two calls to a random function. Hence, from the PRP-PRF Switching Lemma, we have

$$\left| \text{Adv}_{\text{Samvadini}[E]}^{\text{tsprp}}(q, \sigma) - \text{Adv}_{G_1}^{\text{tsprp}}(q, \sigma) \right| \leq \text{Adv}_E^{\text{prp}}(2) + \frac{1}{2^\kappa} .$$

The only difference between G_1 and G_2 is that the single call to $E_{K'_e}$ to generate K'_h is replaced by a single call to π_1 . Further, in each query, the single call to $E_{K'_e}$ is replaced by a single call to π_1 . Hence,

$$\left| \text{Adv}_{G_1}^{\text{tsprp}}(q, \sigma) - \text{Adv}_{G_2}^{\text{tsprp}}(q, \sigma) \right| \leq \text{Adv}_E^{\text{prp}}(q + 1) .$$

The only difference between G_2 and G_3 is that the two E_{K_e} calls to generate K_h and L are replaced by two calls to π_2 . Further, in each query, all the calls to E_{K_e} are replaced by an equal number of calls to π_2 . The total number of calls to E_{K_e} is at most $\sigma + 2$, from which it follows that

$$\left| \text{Adv}_{G_2}^{\text{tsprp}}(q, \sigma) - \text{Adv}_{G_3}^{\text{tsprp}}(q, \sigma) \right| \leq \text{Adv}_E^{\text{prp}}(\sigma + 2) .$$

The result follows from the sum of the individual distances. $\square$

Proposition 8. *The TSPRP advantage of G_3 is bounded as:*

$$\text{Adv}_{G_3}^{\text{tsprp}}(q, \sigma) \leq \text{Adv}_{\mathcal{H}}^{\text{coll}}(q) + \frac{4\sigma^2 + 5q\sigma + 5q^2 + 7\sigma + 2q + 1}{2^n} .$$

Proof. We bound $\text{Adv}_{G_3}^{\text{tsprp}}(q, \sigma)$ using the H-coefficient technique. It allows us to bound the advantage of a deterministic and computationally unbounded distinguisher $\mathcal{A}$ in distinguishing whether a transcript γ , a collection of the queries from $\mathcal{A}$ and the corresponding responses from its oracle, stemmed from the interaction of $\mathcal{A}$ with a real oracle $\mathcal{O}_{\text{re}}$ or with

an ideal oracle $\mathcal{O}_{\text{id}}$. Let Θ_{re} and Θ_{id} denote random variables induced by the interaction of $\mathcal{A}$ with the real and the ideal oracle, respectively. A transcript γ is called attainable with respect to $\mathcal{A}$'s queries if its ideal interpolation probability is nonzero. We assume that the adversary does not make any redundant queries. Then, it follows that

$$\text{Adv}_{G_3}^{\text{tsprp}}(q, \sigma) := \max_{\mathcal{A}} \{ \text{Adv}_{G_3}^{\text{tsprp}}(\mathcal{A}) \} \quad ,$$

where the right-hand side is the maximum over all distinguishers $\mathcal{A}$ that ask at most q queries of at most σ blocks in total. We can use the fundamental result of the H-coefficient technique in the theorem below, the proof of which can be found, e.g., in [Pat08, JN22].

Theorem 6. *Let Ω denote the set of all attainable transcripts and let $\Omega_{\text{bad}} \subseteq \Omega$ denote a set of so-called bad transcripts and its complement $\Omega_{\text{good}} = \Omega \setminus \Omega_{\text{bad}}$ be the set of good transcripts. If it holds that*

$$\Pr[\Theta_{\text{id}} \in \Omega_{\text{bad}}] \leq \epsilon_1$$

and for any transcript $\gamma \in \Omega_{\text{good}}$ that

$$\frac{\Pr[\Theta_{\text{re}} = \gamma]}{\Pr[\Theta_{\text{id}} = \gamma]} \geq 1 - \epsilon_2 \quad ,$$

then the advantage of any deterministic computationally unbounded adversary $\mathcal{A}$ in distinguishing between $\mathcal{O}_{\text{re}}$ and $\mathcal{O}_{\text{id}}$ is upper bounded by $\epsilon_1 + \epsilon_2$.

The online query phase. We start by defining how the G_3 oracle behaves in the real world and in the ideal world during online queries.

- In the real world, all encryption and decryption queries are responded to honestly using the construction G_3 .
- Now, if in the real world P^i is parsed as $p^i \| P_0^i \| \dots \| P_{\ell_i-1}^i$ (resp., C^i is parsed as $c^i \| C_0^i \| \dots \| C_{\ell_i-1}^i$), then the adversary also releases the full output of $\pi_1(Z^i)$. If $|P^i| = |C^i| > 2n$, then it also releases $\pi_2(Z^i \oplus L \oplus \text{bin}(\ell_i - 2))$.
- In the ideal world, the oracle takes a length-preserving tweakable random permutation Π . For each encryption (resp., decryption) query of the form A^i, P^i (resp., A^i, C^i) we compute $C^i := \Pi(A^i, P^i)$ (resp., $P^i := \Pi^{-1}(A^i, C^i)$) and output we compute it as a response.
- Suppose that in the ideal world P^i is parsed as $p^i \| P_0^i \| \dots \| P_{\ell_i-1}^i$ (resp., C^i is parsed as $c^i \| C_0^i \| \dots \| C_{\ell_i-1}^i$). Then, the oracle samples a random value $\rho^i \xleftarrow{\$} \{0, 1\}^{n-|P^i|}$ and defines $\pi_1(Z^i) = (c^i \oplus p^i) \| \rho^i$. If $|P^i| = |C^i| > 2n$, then it also samples a random $\xi^i \xleftarrow{\$} \{0, 1\}^{n-|P_{\ell_i-1}^i|}$ and defines $\pi_2(Z^i \oplus L \oplus \text{bin}(\ell_i - 2)) = (P_{\ell_i-1}^i \oplus C_{\ell_i-1}^i) \| \xi^i$.

Finally, for $T^i := \mathcal{H}(A^i)$, the oracle releases the extended transcript

$$\gamma := \{(P^i, T^i, C^i, Z^i, \pi_2(Z^i \oplus L \oplus \text{bin}(\ell_i - 2))) \mid i \in [1, q]\} \cup \{K_h, L, K_h'\} \quad .$$

Note that if $\exists i \neq i' \in [1, q]$ such that $P^i = P^{i'}$ or $C^i = C^{i'}$, $T^i \neq T^{i'}$ but $T^i = T^{i'}$ then the adversary can trivially distinguish the ideal world from the real world as in the real world this will lead to $(P^i, C^i) = (P^{i'}, C^{i'})$ with probability 1 whereas in the ideal world $(P^i, C^i) = (P^{i'}, C^{i'})$ with negligible probability. Hence, to avoid this trivial distinguishability we define an event

BAD0: There exist $i < i' \in [1, q]$ such that $A^i \neq A^{i'}$ but $T^i = T^{i'}$.

Note that $\Pr[\text{BAD0}] \leq \text{Adv}_{\mathcal{H}}^{\text{coll}}(q)$.

Offline Computations in the Ideal World and BAD Transcripts. After all encryption/decryption queries by the adversary have been asked in the ideal world, the challenger can perform offline computations as follows:

- In the ideal world, given a transcript γ , the challenger initializes two sets $\mathcal{P}_1$ and $\mathcal{P}_2$ as follows:

$$\begin{cases} \mathcal{P}_1 := \{(\text{bin}(0), K_h), (\text{bin}(1), L)\} \\ \mathcal{P}_2 := \{(\text{bin}(0), K'_h)\} \end{cases} .$$

- Then for each $i \in [1, q]$, the challenger parses P^i, C^i as follows:

$$\left\{ \begin{array}{l} \tau := \lceil A^i \rceil_\nu ; \\ P^i := p^i \| P_0^i \| \cdots \| P_{\ell_i-1}^i \text{ s.t. } \ell_i = \begin{cases} 1 & \text{if } |P^i| \leq 2n \\ \left\lceil \frac{|P^i| - \tau}{n} \right\rceil & \text{otherwise} \end{cases} , \\ \text{where:} \\ |P_j^i| = n \text{ for all } j \in [0, \ell_i - 2] \text{ and} \\ |P_{\ell_i-1}^i| = |P^i| - \tau - n(\ell_i - 1) \text{ for } \ell_i \geq 2 ; \text{ and} \\ C^i := c^i \| C_0^i \| \cdots \| C_{\ell_i-1}^i , \\ \text{where:} \\ |C_j^i| = n \text{ for all } j \in [0, \ell_i - 2] \text{ and} \\ |C_{\ell_i-1}^i| = |C^i| - \tau - n(\ell_i - 1) \text{ for } \ell_i \geq 2 . \end{array} \right.$$

- Finally, for each $i \in [1, q]$, the challenger extends $\mathcal{P}_1$ and $\mathcal{P}_2$ as

$$\begin{cases} \mathcal{P}_1 := \{(X^i, Y^i)\} \cup \{(S^i \oplus \text{bin}(j-1), W_j^i) \mid j \in [1, \ell_i - 1]\} \cup \mathcal{P}_1 , \\ \mathcal{P}_2 := \{(Z^i, V^i)\} \cup \mathcal{P}_2 , \end{cases}$$

where

$$\left\{ \begin{array}{l} X^i := P_0^i \oplus H_{K'_h}(\Lambda, p^i) \oplus H_{K'_h}(0^{|p^i|}) \oplus H_{K_h}(T^i, P_1^i \| \cdots \| P_{\ell_i-1}^i) \\ Y^i := C_0^i \oplus H_{K'_h}(\Lambda, c^i) \oplus H_{K_h}(T^i, C_1^i \| \cdots \| C_{\ell_i-1}^i) \\ Z^i := X^i \oplus Y^i \\ V^i := (p^i \oplus c^i) \| \rho^i \\ S^i := Z^i \oplus L \\ W_j^i := \begin{cases} P_j^i \oplus C_j^i & \text{if } j \in [1, \ell_i - 2] \\ (P_{\ell_i-1}^i \oplus C_{\ell_i-1}^i) \| \xi^i & \text{if } j = \ell_i - 1 . \end{cases} \end{array} \right.$$

Definition 12. Let $\mathcal{B}$ be a non-empty set. For any $q \in \mathbb{N}$ let $\mathcal{Q} = \{(X_1, Y_1), \dots, (X_q, Y_q)\} \in (\mathcal{B}^2)^q$ be a collection of q pairs from $\mathcal{B}^2$ each. We call $\mathcal{Q}$ *permutation-compatible* if and only if $X_i = X_j \Leftrightarrow Y_i = Y_j$ holds.

We define some bad events that may help the adversary in distinguishing between the real world and the ideal world:

BAD1: $\mathcal{P}_1$ is not permutation-compatible.

BAD2: $\mathcal{P}_2$ is not permutation-compatible.

Define

$$\text{BAD} := \text{BAD0} \cup \text{BAD1} \cup \text{BAD2} .$$

We call a transcript γ good if BAD does not occur in γ .

Lemma 2.

$$\Pr[\text{BAD}] \leq \text{Adv}_{\mathcal{H}}^{\text{coll}}(q) + \frac{3\sigma^2 + 5q\sigma + 5q^2 + 7\sigma + 2q + 1}{2^n}.$$

This lemma is proved in Appendix C below. Before that, we see how it is used to prove Proposition 8.

Analysis of Good Transcripts. Consider any good transcript γ . Let $\eta(A, m)$ denote the number of queries with tweak A and of length m blocks. Then it must be $\sum_{A, m} \eta(A, m) = \sigma$.

Since $m \geq n$, $\{(A^i, P^i, C^i)\}_{i \in [1, q]}$ has randomness at least $\prod_{A, m} (2^m)^{\eta(A, m)} \geq (2^n - \sigma)^\sigma$.

Given any $\{(A^i, P^i, C^i)\}_{i \in [1, q]}$, let the i -th query have ℓ_i blocks. For any $\tau \in [0, n]$, note that $|P_{\ell_i-1}^i| = |P^i| - (\ell_i - 1) \cdot n - \tau$. Hence, in the ideal world $W_{\ell_i-1}^i$ (i.e., ξ^i) has $\ell_i \cdot n - |P^i| + \tau$ random independent bits. Similarly, V^i (i.e., ρ^i) $n - \tau$ random bits. Hence, the extended transcript $(W_{\ell_i-1}^i, V^i)$ consists of $(\ell_i + 1) \cdot n - |P^i|$ random bits. Since by definition of ℓ_i , $|P^i| \leq \ell_i \cdot n$, the extended transcript has at least n random bits. Being this true for any $i \in [1, q]$ and $\tau \in [0, n]$, and K_h, K'_h, L are also sampled uniformly at random, we have

$$\Pr[\Theta_{\text{id}} = \tau] \leq \frac{1}{2^{3n+qn}} \times \frac{1}{(2^n - \sigma)^\sigma}.$$

In the real world, the entire transcript γ has been generated with the use of π_1 and π_2 . For each query by the adversary to its oracle, there is exactly one call to π_1 and at most ℓ_i calls to π_2 . Further, K_h and L are generated using queries to π_2 , and K'_h is generated using a query to π_1 . Thus

$$\Pr[\Theta_{\text{re}} = \tau] \geq \frac{1}{(2^n)_{q+1}} \times \frac{1}{(2^n)_{\sigma+2}} \geq \frac{1}{2^{n(\sigma+q+3)}}.$$

It is now easy to see that, given any good transcript γ ,

$$\frac{\Pr[\Theta_{\text{re}} = \gamma]}{\Pr[\Theta_{\text{id}} = \gamma]} \geq 1 - \frac{\sigma^2}{2^n}.$$

Hence, by the H-coefficient technique, Lemma 2 implies Proposition 8. $\square$

C Proof of Lemma 2

Proof. In this section we will bound $\Pr[\text{BAD1}|\overline{\text{BAD0}}]$ and $\Pr[\text{BAD2}|\overline{\text{BAD0}} \cup \text{BAD1}]$. Lemma 2 then follows from the union bound.

Lemma 3. *It is*

$$\delta(\Lambda, p^i) = \delta(\Lambda, c^i) \leq 2 \quad \text{and} \quad \sum_{i \in [1, q]} \delta(T^i, P_1^i) \cdots \|P_{\ell_i-1}^i\| \leq \sigma.$$

Proof. Since $|p^i|, |c^i| \leq n$ and $|P_0^i| = n$, the result follows from Proposition 1. $\square$

C.1 Bounding $\Pr[\text{BAD1}|\overline{\text{BAD0}}]$

We start by recalling the definition of the set $\mathcal{P}_1$:

$$\begin{aligned} \mathcal{P}_1 = & \{(\text{bin}(0), K_h), (\text{bin}(1), L)\} \cup \\ & \cup \bigcup_{i \in [1, q]} \{(X^i, Y^i), (S^i \oplus \text{bin}(0), W_1^i), \dots, (S^i \oplus \text{bin}(\ell_i - 2), W_{\ell_i-1}^i)\}. \end{aligned}$$

Note that if for all $(A, B) \neq (A', B') \in \mathcal{P}_1$, $A \neq A'$ and $B \neq B'$ then $\mathcal{P}_1$ is permutation-compatible. Hence

$$\begin{aligned} \Pr[\text{BAD1}] &\leq \Pr[\exists (A, B) \neq (A', B') \in \mathcal{P}_1 \mid A = A'] \\ &\quad + \Pr[\exists (A, B) \neq (A', B') \in \mathcal{P}_1 \mid B = B'] . \end{aligned}$$

We consider the two probabilities on the r.h.s. separately

C.1.1 Bounding $\Pr[\exists (A, B) \neq (A', B') \in \mathcal{P}_1 \mid A = A']$

This case can be divided into the following sub-cases:

B1: There exists an $i \in [1, q]$ such that $X^i \in \{\text{bin}(0), \text{bin}(1)\}$.

This happens if and only if

$$P_0^i \oplus H_{K'_h}(\Lambda, p^i) \oplus H_{K'_h}(0^{|p|}) \oplus H_{K_h}(T^i, P_1^i \parallel \cdots \parallel P_{\ell_i-1}^i) \in \{\text{bin}(0), \text{bin}(1)\}$$

for some $i \in [1, q]$. Since K_h is uniformly sampled, at the end of all encryption/decryption queries, by [Lemma 3](#) we have:

$$\begin{aligned} \Pr[\text{B1}] &\leq \sum_{i \in [1, q]} \Pr \left[\begin{array}{l} H_{K_h}(T^i, P_1^i \parallel \cdots \parallel P_{\ell_i-1}^i) = a \mid \\ P_0^i \oplus H_{K'_h}(\Lambda, p^i) \oplus H_{K'_h}(0^{|p|}) = a \oplus \text{bin}(0) \end{array} \right] \\ &\quad + \sum_{i \in [1, q]} \Pr \left[\begin{array}{l} H_{K_h}(T^i, P_1^i \parallel \cdots \parallel P_{\ell_i-1}^i) = a \mid \\ P_0^i \oplus H_{K'_h}(\Lambda, p^i) \oplus H_{K'_h}(0^{|p|}) = a \oplus \text{bin}(1) \end{array} \right] \\ &\leq \sum_{i \in [1, q]} \frac{\delta(T^i, P_1^i \parallel \cdots \parallel P_{\ell_i-1}^i)}{2^n} \leq \frac{\sigma}{2^n} . \end{aligned}$$

B2: There exist $i \in [1, q]$, $j \in [1, \ell_i - 1]$ with $S^i \oplus \text{bin}(j - 1) \in \{\text{bin}(0), \text{bin}(1)\}$.

This happens if and only if for some $i \in [1, q]$

$$L = X^i \oplus Y^i \oplus \text{bin}(j - 1) \in \{\text{bin}(0), \text{bin}(1)\} .$$

Note that X^i, Y^i are calculated independently from L . Since L is generated uniformly at random at the end of all queries, we have

$$\Pr[\text{B2}] \leq \frac{2\sigma}{2^n} .$$

B3: There exist $i < i' \in [1, q]$ such that $X^i = X^{i'}$.

We split the determination of the bound into three separate contributions:

- (i) First suppose $p^i \neq p^{i'}$, then from the AXU property of $H_{K'_h}$ and by [Proposition 2](#) and [Lemma 3](#), the probability that this event happens for some $i < i'$ is bounded by $2/2^n$.
- (ii) Next suppose $p^i = p^{i'}$ and $(T^i, P_1^i \parallel \cdots \parallel P_{\ell_i-1}^i) \neq (T^{i'}, P_1^{i'} \parallel \cdots \parallel P_{\ell_{i'}-1}^{i'})$, then from the AXU property of H_{K_h} and by [Proposition 2](#), the probability that this happens for some $i < i'$ is bounded by

$$\frac{\max\{\delta(T^i, P_1^i \parallel \cdots \parallel P_{\ell_i-1}^i), \delta(T^{i'}, P_1^{i'} \parallel \cdots \parallel P_{\ell_{i'}-1}^{i'})\}}{2^n} .$$

- (iii) Finally if $p^i = p^{i'}$ and $(T^i, P_1^i \parallel \dots \parallel P_{\ell_i-1}^i) = (T^{i'}, P_1^{i'} \parallel \dots \parallel P_{\ell_{i'}-1}^{i'})$ then since the event BAD0 does not occur, this happens for $i < j$ if and only if i' is a decryption query, and $P_0^i = P_0^{i'}$ which is impossible since Π is defined as a length-preserving tweakable random permutation.

Hence

$$\begin{aligned}
\Pr[\text{B3}] &\leq \sum_{i < i'} \frac{2}{2^n} + \sum_{i < i'} \frac{\max\{\delta(T^i, P_1^i \parallel \dots \parallel P_{\ell_i-1}^i), \delta(T^{i'}, P_1^{i'} \parallel \dots \parallel P_{\ell_{i'}-1}^{i'})\}}{2^n} \\
&\leq \frac{2q^2}{2^n} + \sum_{i < i'} \frac{\delta(T^i, P_1^i \parallel \dots \parallel P_{\ell_i-1}^i)}{2^n} + \sum_{i < i'} \frac{\delta(T^{i'}, P_1^{i'} \parallel \dots \parallel P_{\ell_{i'}-1}^{i'})}{2^n} \\
&\leq \frac{2q^2}{2^n} + \frac{q}{2^n} \left(\sum_{i \in [1, q]} \delta(T^i, P_1^i \parallel \dots \parallel P_{\ell_i-1}^i) + \sum_{i' \in [1, q]} \delta(T^{i'}, P_1^{i'} \parallel \dots \parallel P_{\ell_{i'}-1}^{i'}) \right) \\
&\leq \frac{2q^2}{2^n} + \frac{2q\sigma}{2^n} .
\end{aligned}$$

The last inequality follows from [Lemma 3](#).

- B4: There exist $i < i' \in [1, q]$ and j, j' with $j \in [1, \ell_i - 1]$ and $j' \in [1, \ell_{i'} - 1]$ such that $S^i \oplus \text{bin}(j - 1) = S^{i'} \oplus \text{bin}(j' - 1)$.

Note that this happens if and only if

$$C_0^i \oplus C_0^{i'} \oplus P_0^i \oplus P_0^{i'} = \text{bin}(j - 1) \oplus \text{bin}(j' - 1) \oplus a$$

where a depends from the values $K_h, K'_h, c^i, c^{i'}, p^i, p^{i'}, (T^i, P_1^i \parallel \dots \parallel P_{\ell_i-1}^i)$ as well as $(T^{i'}, P_1^{i'} \parallel \dots \parallel P_{\ell_{i'}-1}^{i'})$, but not from $C_0^i \oplus C_0^{i'} \oplus P_0^i \oplus P_0^{i'}$.

Hence, if i' is an encryption (resp., decryption) query, then using the randomness of $C_0^{i'}$ (resp., $P_0^{i'}$) and varying over all $i < i'$ and j, j' we have

$$\Pr[\text{B4}] \leq \frac{0.5\sigma^2}{2^n} .$$

- B5: There exist $i, i' \in [1, q]$ and $j \in [1, \ell_i - 1]$ such that $S^i \oplus \text{bin}(j - 1) = X^{i'}$.

- If $i < i'$, suppose $S^i = a$. Then this event is similar to the event $X^{i'} \in \{a \oplus \text{bin}(j - 1) \mid j \in [1, \ell_i - 1]\}$. Doing a similar analysis as B1, We have

$$\Pr[X^{i'} = S^i \oplus \text{bin}(j - 1) \mid i < i'] \leq \frac{\ell_i \delta(T^{i'}, P_1^{i'}, \dots, P_{\ell_{i'}-1}^{i'})}{2^n} .$$

- Otherwise suppose $i' < i$, suppose $X^{i'} = a$. Then this event is similar to the event $\exists j \in [1, \ell_i - 1] \text{ s.t. } S^i \oplus \text{bin}(j - 1) = a$. By a similar analysis as for B2, we obtain

$$\Pr[X^{i'} = S^i \oplus \text{bin}(j - 1) \mid i' < i] \leq \frac{\ell_i}{2^n} .$$

Hence varying over all i, i' we have

$$\Pr[\text{B5}] \leq \frac{\sigma^2}{2^n} + \frac{q\sigma}{2^n} .$$

C.1.2 Bounding $\Pr [\exists (A, B) \neq (A', B') \in \mathcal{P}_1 \mid B = B']$

This case can be divided into the following sub-cases:

B6: It is $K_h = L$.

Since both K_h, L are independently generated, we have

$$\Pr [\text{B6}] \leq \frac{1}{2^n} .$$

B7: There exists an $i \in [1, q]$ such that $Y^i \in \{K_h, L\}$.

Following the same analysis as **B1**, we have

$$\begin{aligned} \Pr [\text{B7}] &\leq \sum_{i \in [1, q]} \Pr \left[H_{K_h}(T^i, C_1^i \parallel \dots \parallel C_{\ell_i-1}^i) = a \mid C_0^i \oplus H_{K_h'}(\Lambda, c^i) = a \right] \\ &+ \sum_{i \in [1, q]} \Pr \left[H_{K_h}(T^i, C_1^i \parallel \dots \parallel C_{\ell_i-1}^i) = a \mid C_0^i \oplus H_{K_h'}(\Lambda, c^i) = a \oplus L \right] . \end{aligned}$$

Now note that K_h, L are generated uniformly at random at the end of all the queries. So, by using Propositions 1 and 3, we have

$$\Pr [\text{B7}] \leq \frac{\sigma + q}{2^n} .$$

B8: There exist $i \in [1, q]$ and $j \in [1, \ell_i - 1]$ such that $W_j^i \in \{K_h, L\}$.

Note that W_j^i are completely determined by the transcript. Also, K_h, L are independently generated in the transcript. Hence

$$\Pr [\text{B8}] \leq \frac{2\sigma}{2^n} .$$

B9: There exist $i < i' \in [1, q]$ such that $Y^i = Y^{i'}$.

With a similar analysis as for **B3**, we have

$$\Pr [\text{B9}] \leq \frac{2q^2 + q\sigma}{2^n} .$$

B10: There exist $i < i' \in [1, q]$, $j \in [1, \ell_i - 1]$, $j' \in [1, \ell_{i'} - 1]$ with $W_j^i = W_{j'}^{i'}$.

Note that W_j^i are completely determined from the transcript. Hence,

$$\Pr [\text{B10}] \leq \frac{0.5\sigma^2}{2^n} .$$

B11: There exist $i, i' \in [1, q]$, $j \in [1, \ell_i - 1]$ such that $W_j^i = Y^{i'}$.

With a similar analysis as for **B5**, we get

$$\Pr [\text{B11}] \leq \frac{\sigma^2}{2^n} + \frac{q\sigma}{2^n} .$$

C.2 Bounding $\Pr [\text{BAD2} | \overline{\text{BAD1} \cup \text{BADO}}]$

First of all, recall that $\mathcal{P}_2 = \{(\text{bin}(0), K'_h)\} \cup \bigcup_{i \in [1, q]} \{(Z^i, V^i)\}$. Note that, given **BAD1** and **BADO** do not occur, we have for all $i \neq i'$ that $S^i \neq S^{i'}$. Hence, by definition $Z^i \neq Z^{i'}$. In a similar way to the case of **BAD1**, we can thus divide this event into the following subcases:

B12: There exists an $i \in [1, q]$ such that $Z^i = \text{bin}(0)$.

Note that this event is equivalent to the event that $\exists i \in [1, q]$ s.t. $S_i = L$. which is equivalent to the event that $X^i \oplus Y^i = 0$. Now note that, depending on whether it is an encryption (resp., decryption) query, this event is bounded by the randomness of X^i (resp., Y^i). Hence,

$$\Pr [\text{B12}] \leq \frac{\sigma}{2^n} .$$

B13: There exists an $i \in [1, q]$ such that $V^i = K'_h$.

Note that for any i , the two values V^i and K'_h are independently generated in the transcript. Hence,

$$\Pr [\text{B13}] \leq \frac{q}{2^n} .$$

B14: There exist $i < i' \in [1, q]$ such that $V^i = V^{i'}$.

Note that for all i , the V^i are independently generated. Hence

$$\Pr [\text{B14}] \leq \frac{q^2}{2^n} .$$

Finally, using the Encode-then-Encipher approach with the **Encode** function defined in [Subsection 6.1](#) and the enciphering scheme **Samvadini**, [Theorem 4](#) follows immediately from [Lemma 1](#), [Proposition 8](#), and [Theorem 1](#). $\square$

References

- [ABD⁺23] Roberto Avanzi, Subhadeep Banik, Orr Dunkelman, Maria Eichlseder, Shibam Ghosh, Marcel Nageler, and Francesco Regazzoni. The QARMAv2 Family of Tweakable Block Ciphers. *IACR Transactions on Symmetric Cryptology*, 3:25–73, 9 2023. doi:10.46586/tosc.v2023.i3.25-73.
- [ABL⁺14] Elena Andreeva, Andrey Bogdanov, Atul Luykx, Bart Mennink, Nicky Mouha, and Kan Yasuda. How to Securely Release Unverified Plaintext in Authenticated Encryption. In Palash Sarkar and Tetsu Iwata, editors, *Advances in Cryptology - ASIACRYPT 2014 - 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, R.O.C., December 7-11, 2014. Proceedings, Part I*, volume 8873 of *Lecture Notes in Computer Science*, pages 105–125. Springer, 2014. doi:10.1007/978-3-662-45611-8_6.
- [ABN10] Michel Abdalla, Mihir Bellare, and Gregory Neven. Robust Encryption. In Daniele Micciancio, editor, *Theory of Cryptography, 7th Theory of Cryptography Conference, TCC 2010, Zurich, Switzerland, February 9-11, 2010. Proceedings*, volume 5978 of *Lecture Notes in Computer Science*, pages 480–497. Springer, 2010. doi:10.1007/978-3-642-11799-2_28.

- [ABPV21] Elena Andreeva, Amit Singh Bhati, Bart Preneel, and Damian Vizár. 1, 2, 3, Fork: Counter Mode Variants based on a Generalized Forkcipher. *IACR Trans. Symmetric Cryptol.*, 2021(3):1–35, 2021. doi:[10.46586/TOSC.V2021.I3.1-35](https://doi.org/10.46586/TOSC.V2021.I3.1-35).
- [ADG⁺22] Ange Albertini, Thai Duong, Shay Gueron, Stefan Kölbl, Atul Luykx, and Sophie Schmieg. How to Abuse and Fix Authenticated Encryption Without Key Commitment. In Kevin R. B. Butler and Kurt Thomas, editors, *31st USENIX Security Symposium, USENIX Security 2022, Boston, MA, USA, August 10-12, 2022*, pages 3291–3308. USENIX Association, 2022. Available from: <https://www.usenix.org/conference/usenixsecurity22/presentation/albertini>.
- [ADM24] Roberto Avanzi, Orr Dunkelman, and Kazuhiko Minematsu. MATTER: A Wide-Block Tweakable Block Cipher. *IACR Cryptol. ePrint Arch.*, page 1186, 2024. Available from: <https://eprint.iacr.org/2024/1186>.
- [ANWW13] Jean-Philippe Aumasson, Samuel Neves, Zooko Wilcox-O’Hearn, and Christian Winnerlein. BLAKE2: Simpler, Smaller, Fast as MD5. In Michael J. Jacobson Jr., Michael E. Locasto, Payman Mohassel, and Reihaneh Safavi-Naini, editors, *Applied Cryptography and Network Security - 11th International Conference, ACNS 2013, Banff, AB, Canada, June 25-28, 2013. Proceedings*, volume 7954 of *Lecture Notes in Computer Science*, pages 119–135. Springer, 2013. doi:[10.1007/978-3-642-38980-1_8](https://doi.org/10.1007/978-3-642-38980-1_8).
- [BB93] Ishai Ben-Aroya and Eli Biham. Differential Cryptanalysis of Lucifer. In Douglas R. Stinson, editor, *Advances in Cryptology - CRYPTO ’93, 13th Annual International Cryptology Conference, Santa Barbara, California, USA, August 22-26, 1993, Proceedings*, volume 773 of *Lecture Notes in Computer Science*, pages 187–199. Springer, 1993. doi:[10.1007/3-540-48329-2_17](https://doi.org/10.1007/3-540-48329-2_17).
- [BBD⁺24] Arghya Bhattacharjee, Ritam Bhaumik, Nilanjan Datta, Avijit Dutta, and Sougata Mandal. BBB Secure Arbitrary Length Tweak TBC from n-bit Block Ciphers. *IACR Cryptol. ePrint Arch.*, page 2049, 2024. Available from: <https://eprint.iacr.org/2024/2049>.
- [Ber07] Daniel J. Bernstein. Polynomial evaluation and message authentication. *A manuscript*, October 2007. Available from: <https://cr.yp.to/papers.html#pema>.
- [BH22] Mihir Bellare and Viet Tung Hoang. Efficient Schemes for Committing Authenticated Encryption. In Orr Dunkelman and Stefan Dziembowski, editors, *Advances in Cryptology - EUROCRYPT 2022 - 41st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Trondheim, Norway, May 30 - June 3, 2022, Proceedings, Part II*, volume 13276 of *Lecture Notes in Computer Science*, pages 845–875. Springer, 2022. doi:[10.1007/978-3-031-07085-3_29](https://doi.org/10.1007/978-3-031-07085-3_29).
- [BH24] Mihir Bellare and Viet Tung Hoang. Succinctly-Committing Authenticated Encryption. In Leonid Reyzin and Douglas Stebila, editors, *Advances in Cryptology - CRYPTO 2024 - 44th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2024, Proceedings, Part IV*, volume 14923 of *Lecture Notes in Computer Science*, pages 305–339. Springer, 2024. doi:[10.1007/978-3-031-68385-5_10](https://doi.org/10.1007/978-3-031-68385-5_10).

- [BHI⁺23] Zhenzhen Bao, Seongha Hwang, Akiko Inoue, ByeongHak Lee, Jooyoung Lee, and Kazuhiko Minematsu. XOCB: Beyond-Birthday-Bound Secure Authenticated Encryption Mode with Rate-One Computation. In Carmit Hazay and Martijn Stam, editors, *Advances in Cryptology - EUROCRYPT 2023 - 42nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Lyon, France, April 23-27, 2023, Proceedings, Part IV*, volume 14007 of *Lecture Notes in Computer Science*, pages 532–561. Springer, 2023. doi:10.1007/978-3-031-30634-1_18.
- [Bla85] Richard E. Blahut. *Fast Algorithms for Signal Processing*. Addison-Wesley Publishing Company, 1985.
- [Bla10] Richard E. Blahut. *Fast Algorithms for Signal Processing*. Cambridge University Press, 2010.
- [BLLS22] Jannis Bossert, Eik List, Stefan Lucks, and Sebastian Schmitz. Pholkos - Efficient Large-State Tweakable Block Ciphers from the AES Round Function. In Steven D. Galbraith, editor, *Topics in Cryptology - CT-RSA 2022 - Cryptographers’ Track at the RSA Conference 2022, Virtual Event, March 1-2, 2022, Proceedings*, volume 13161 of *Lecture Notes in Computer Science*, pages 511–536. Springer, 2022. doi:10.1007/978-3-030-95312-6_21.
- [BLN18] Ritam Bhaumik, Eik List, and Mridul Nandi. ZCZ - Achieving n-bit SPRP Security with a Minimal Number of Tweakable-Block-Cipher Calls. In Thomas Peyrin and Steven D. Galbraith, editors, *Advances in Cryptology - ASIACRYPT 2018 - 24th International Conference on the Theory and Application of Cryptology and Information Security, Brisbane, QLD, Australia, December 2-6, 2018, Proceedings, Part I*, volume 11272 of *Lecture Notes in Computer Science*, pages 336–366. Springer, 2018. doi:10.1007/978-3-030-03326-2_12.
- [Blo24] Blockchain.com. Total Hash Rate, 2024. Available from: <https://www.blockchain.com/explorer/charts/hash-rate>.
- [BMM⁺15] Christian Badertscher, Christian Matt, Ueli Maurer, Phillip Rogaway, and Björn Tackmann. Robust Authenticated Encryption and the Limits of Symmetric Cryptography. In Jens Groth, editor, *Cryptography and Coding - 15th IMA International Conference, IMACC 2015, Oxford, UK, December 15-17, 2015. Proceedings*, volume 9496 of *Lecture Notes in Computer Science*, pages 112–129. Springer, 2015. doi:10.1007/978-3-319-27239-9_7.
- [BR00] Mihir Bellare and Phillip Rogaway. Encode-Then-Encipher Encryption: How to Exploit Nonces or Redundancy in Plaintexts for Efficient Cryptography. In Tatsuaki Okamoto, editor, *Advances in Cryptology - ASIACRYPT 2000, 6th International Conference on the Theory and Application of Cryptology and Information Security, Kyoto, Japan, December 3-7, 2000, Proceedings*, volume 1976 of *Lecture Notes in Computer Science*, pages 317–330. Springer, 2000. doi:10.1007/3-540-44448-3_24.
- [BVA24] Amit Singh Bhati, Michiel Verbauwhede, and Elena Andreeva. Breaking, Repairing and Enhancing XCBv2 into the Tweakable Enciphering Mode GEM. Cryptology ePrint Archive, Paper 2024/1554, 2024. Available from: <https://eprint.iacr.org/2024/1554>.
- [CB18] Paul Crowley and Eric Biggers. Adiantum: length-preserving encryption for entry-level processors. *IACR Trans. Symmetric Cryptol.*, 2018(4):39–61, 2018. doi:10.13154/TOSC.V2018.I4.39-61.

- [CDL⁺20] Anne Canteaut, Sébastien Duval, Gaëtan Leurent, María Naya-Plasencia, Léo Perrin, Thomas Pornin, and André Schrottenloher. Saturnin: a suite of lightweight symmetric algorithms for post-quantum security. *IACR Trans. Symmetric Cryptol.*, 2020(S1):160–207, 2020. doi:[10.13154/TOSC.V2020.IS1.160-207](https://doi.org/10.13154/TOSC.V2020.IS1.160-207).
- [CEL⁺21] Benoît Cogliati, Jordan Ethan, Virginie Lallemand, ByeongHak Lee, Jooyoung Lee, and Marine Minier. CTET+: A Beyond-Birthday-Bound Secure Tweakable Enciphering Scheme Using a Single Pseudorandom Permutation. *IACR Trans. Symmetric Cryptol.*, 2021(4):1–35, 2021. doi:[10.46586/TOSC.V2021.I4.1-35](https://doi.org/10.46586/TOSC.V2021.I4.1-35).
- [CFI⁺23] Yu Long Chen, Antonio Flórez-Gutiérrez, Akiko Inoue, Ryoma Ito, Tetsu Iwata, Kazuhiko Minematsu, Nicky Mouha, Yusuke Naito, Ferdinand Sibleyras, and Yosuke Todo. Key Committing Security of AEZ and More. *IACR Trans. Symmetric Cryptol.*, 2023(4):452–488, 2023. doi:[10.46586/TOSC.V2023.I4.452-488](https://doi.org/10.46586/TOSC.V2023.I4.452-488).
- [CGLS22] Debrup Chakraborty, Sebati Ghosh, Cuauhtemoc Mancillas López, and Palash Sarkar. FAST: Disk encryption and beyond. *Adv. Math. Commun.*, 16(1):185–230, 2022. doi:[10.3934/AMC.2020108](https://doi.org/10.3934/AMC.2020108).
- [CHB21] Paul Crowley, Nathan Huckleberry, and Eric Biggers. Length-preserving encryption with HCTR2. *IACR Cryptol. ePrint Arch.*, page 1441, 2021. Available from: <https://eprint.iacr.org/2021/1441>.
- [CLNY06] Donghoon Chang, Sangjin Lee, Mridul Nandi, and Moti Yung. Indifferentiable Security Analysis of Popular Hash Functions with Prefix-Free Padding. In Xuejia Lai and Kefei Chen, editors, *Advances in Cryptology - ASIACRYPT 2006, 12th International Conference on the Theory and Application of Cryptology and Information Security, Shanghai, China, December 3-7, 2006, Proceedings*, volume 4284 of *Lecture Notes in Computer Science*, pages 283–298. Springer, 2006. doi:[10.1007/11935230_19](https://doi.org/10.1007/11935230_19).
- [CN08a] Debrup Chakraborty and Mridul Nandi. An Improved Security Bound for HCTR. In Kaisa Nyberg, editor, *Fast Software Encryption, 15th International Workshop, FSE 2008, Lausanne, Switzerland, February 10-13, 2008, Revised Selected Papers*, volume 5086 of *Lecture Notes in Computer Science*, pages 289–302. Springer, 2008. doi:[10.1007/978-3-540-71039-4_18](https://doi.org/10.1007/978-3-540-71039-4_18).
- [CN08b] Donghoon Chang and Mridul Nandi. Improved Indifferentiability Security Analysis of chopMD Hash Function. In Kaisa Nyberg, editor, *Fast Software Encryption, 15th International Workshop, FSE 2008, Lausanne, Switzerland, February 10-13, 2008, Revised Selected Papers*, volume 5086 of *Lecture Notes in Computer Science*, pages 429–443. Springer, 2008. doi:[10.1007/978-3-540-71039-4_27](https://doi.org/10.1007/978-3-540-71039-4_27).
- [CS06a] Debrup Chakraborty and Palash Sarkar. HCH: A New Tweakable Enciphering Scheme Using the Hash-Encrypt-Hash Approach. In Rana Barua and Tanja Lange, editors, *Progress in Cryptology - INDOCRYPT 2006, 7th International Conference on Cryptology in India, Kolkata, India, December 11-13, 2006, Proceedings*, volume 4329 of *Lecture Notes in Computer Science*, pages 287–302. Springer, 2006. doi:[10.1007/11941378_21](https://doi.org/10.1007/11941378_21).
- [CS06b] Debrup Chakraborty and Palash Sarkar. A New Mode of Encryption Providing a Tweakable Strong Pseudo-random Permutation. In Matthew J. B.

- Robshaw, editor, *Fast Software Encryption, 13th International Workshop, FSE 2006, Graz, Austria, March 15-17, 2006, Revised Selected Papers*, volume 4047 of *Lecture Notes in Computer Science*, pages 293–309. Springer, 2006. doi:10.1007/11799313_19.
- [Dam89] Ivan Damgård. A Design Principle for Hash Functions. In Gilles Brassard, editor, *Advances in Cryptology - CRYPTO '89, 9th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 1989, Proceedings*, volume 435 of *Lecture Notes in Computer Science*, pages 416–427. Springer, 1989. doi:10.1007/0-387-34805-0_39.
- [DDGN24] Nilanjan Datta, Avijit Dutta, Shibam Ghosh, and Hrithik Nandi. Optimally Secure TBC Based Accordion Mode. *IACR Cryptol. ePrint Arch.*, page 2053, 2024. Available from: <https://eprint.iacr.org/2024/2053>.
- [DFUB24] Hieu Nguyen Duy, Pablo García Fernández, Aleksei Udovenko, and Alex Biryukov. Accordion mode based on Hash-Encrypt-Hash, June 2024. Available from: <https://csrc.nist.gov/csrc/media/Events/2024/accordion-cipher-mode-workshop-2024/documents/papers/accordion-mode-based-hash-encrypt-hash.pdf>.
- [DG22] Nir Drucker and Shay Gueron. Software Optimization of Rijndael for Modern x86-64 Platforms. In Shahram Latifi, editor, *ITNG 2022 19th International Conference on Information Technology-New Generations*, pages 147–153. Springer International Publishing, 2022. doi:10.1007/978-3-030-97652-1_18.
- [DGGP24] Jean Paul Degabriele, Jan Gilcher, Jérôme Govinden, and Kenneth G. Paterson. Universal Hash Designs for an Accordion Mode (Slides), June 2024. Available from: <https://csrc.nist.gov/csrc/media/Presentations/2024/universal-hash-designs-for-an-accordion-mode/images-media/sess-7-degabriele-acm-workshop-2024.pdf>.
- [DGRW18] Yevgeniy Dodis, Paul Grubbs, Thomas Ristenpart, and Joanne Woodage. Fast Message Franking: From Invisible Salamanders to Encryptment. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology - CRYPTO 2018 - 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2018, Proceedings, Part I*, volume 10991 of *Lecture Notes in Computer Science*, pages 155–186. Springer, 2018. doi:10.1007/978-3-319-96884-1_6.
- [DMMT24] Christoph Dobraunig, Krystian Matusiewicz, Bart Mennink, and Alexander Tereschenko. Efficient Instances of Docked Double Decker With AES. *IACR Cryptol. ePrint Arch.*, page 84, 2024. Available from: <https://eprint.iacr.org/2024/084>.
- [DN18] Avijit Dutta and Mridul Nandi. Tweakable HCTR: A BBB Secure Tweakable Enciphering Scheme. In Debrup Chakraborty and Tetsu Iwata, editors, *Progress in Cryptology - INDOCRYPT 2018 - 19th International Conference on Cryptology in India, New Delhi, India, December 9-12, 2018, Proceedings*, volume 11356 of *Lecture Notes in Computer Science*, pages 47–69. Springer, 2018. doi:10.1007/978-3-030-05378-9_3.
- [DR02] Joan Daemen and Vincent Rijmen. *The Design of Rijndael: AES — The Advanced Encryption Standard*. Information Security and Cryptography. Springer, 2002. doi:10.1007/978-3-662-04722-4.

- [FLS⁺10] Niels Ferguson, Stefan Lucks, Bruce Schneier, Doug Whiting Hifn, Mihir Bellare, Tadayoshi Kohno, Jon Callas, and Jesse Walker. The Skein Hash Function Family, Version 1.3, October 2010.
- [FOR17] Pooya Farshim, Claudio Orlandi, and Razvan Rosie. Security of Symmetric Primitives under Incorrect Usage of Keys. *IACR Trans. Symmetric Cryptol.*, 2017(1):449–473, 2017. doi:[10.13154/TOSC.V2017.I1.449-473](https://doi.org/10.13154/TOSC.V2017.I1.449-473).
- [GDM19] Aldo Gunesing, Joan Daemen, and Bart Mennink. Deck-Based Wide Block Cipher Modes and an Exposition of the Blinded Keyed Hashing Model. *IACR Trans. Symmetric Cryptol.*, 2019(4):1–22, 2019. doi:[10.13154/TOSC.V2019.I4.1-22](https://doi.org/10.13154/TOSC.V2019.I4.1-22).
- [GLL17] Shay Gueron, Adam Langley, and Yehuda Lindell. AES-GCM-SIV: Specification and Analysis. *IACR Cryptology ePrint Archive*, 2017:168, 2017. Available from: <http://eprint.iacr.org/2017/168>.
- [GLR17] Paul Grubbs, Jiahui Lu, and Thomas Ristenpart. Message Franking via Committing Authenticated Encryption. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology - CRYPTO 2017 - 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 20-24, 2017, Proceedings, Part III*, volume 10403 of *Lecture Notes in Computer Science*, pages 66–97. Springer, 2017. doi:[10.1007/978-3-319-63697-9_3](https://doi.org/10.1007/978-3-319-63697-9_3).
- [Hal07] Shai Halevi. Invertible Universal Hashing and the TET Encryption Mode. In Alfred Menezes, editor, *Advances in Cryptology - CRYPTO 2007, 27th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2007, Proceedings*, volume 4622 of *Lecture Notes in Computer Science*, pages 412–429. Springer, 2007. doi:[10.1007/978-3-540-74143-5_23](https://doi.org/10.1007/978-3-540-74143-5_23).
- [HKR15] Viet Tung Hoang, Ted Krovetz, and Phillip Rogaway. Robust Authenticated-Encryption AEZ and the Problem That It Solves. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part I*, volume 9056 of *Lecture Notes in Computer Science*, pages 15–44. Springer, 2015. doi:[10.1007/978-3-662-46800-5_2](https://doi.org/10.1007/978-3-662-46800-5_2).
- [HM24] Viet Tung Hoang and Sanketh Menda. Robust AE With Committing Security. In Kai-Min Chung and Yu Sasaki, editors, *Advances in Cryptology - ASIACRYPT 2024 - 30th International Conference on the Theory and Application of Cryptology and Information Security, Kolkata, India, December 9-13, 2024, Proceedings, Part IX*, volume 15492 of *Lecture Notes in Computer Science*, pages 312–342. Springer, 2024. doi:[10.1007/978-981-96-0947-5_11](https://doi.org/10.1007/978-981-96-0947-5_11).
- [HR03] Shai Halevi and Phillip Rogaway. A Tweakable Enciphering Mode. In Dan Boneh, editor, *Advances in Cryptology - CRYPTO 2003, 23rd Annual International Cryptology Conference, Santa Barbara, California, USA, August 17-21, 2003, Proceedings*, volume 2729 of *Lecture Notes in Computer Science*, pages 482–499. Springer, 2003. doi:[10.1007/978-3-540-45146-4_28](https://doi.org/10.1007/978-3-540-45146-4_28).
- [HR04] Shai Halevi and Phillip Rogaway. A Parallelizable Enciphering Mode. In Tatsuaki Okamoto, editor, *Topics in Cryptology - CT-RSA 2004, The Cryptographers’ Track at the RSA Conference 2004, San Francisco, CA, USA, Febru-*

- ary 23-27, 2004, *Proceedings*, volume 2964 of *Lecture Notes in Computer Science*, pages 292–304. Springer, 2004. doi:[10.1007/978-3-540-24660-2_23](https://doi.org/10.1007/978-3-540-24660-2_23).
- [IMPS17] Tetsu Iwata, Kazuhiko Minematsu, Thomas Peyrin, and Yannick Seurin. ZMAC: A Fast Tweakable Block Cipher Mode for Highly Secure Message Authentication. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology - CRYPTO 2017 - 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 20-24, 2017, Proceedings, Part III*, volume 10403 of *Lecture Notes in Computer Science*, pages 34–65. Springer, 2017. doi:[10.1007/978-3-319-63697-9_2](https://doi.org/10.1007/978-3-319-63697-9_2).
- [IMV16] Tetsu Iwata, Bart Mennink, and Damian Vizár. CENC is Optimally Secure. *IACR Cryptol. ePrint Arch.*, page 1087, 2016. Available from: <http://eprint.iacr.org/2016/1087>.
- [Iwa06] Tetsu Iwata. New Blockcipher Modes of Operation with Beyond the Birthday Bound Security. In Matthew J. B. Robshaw, editor, *Fast Software Encryption, 13th International Workshop, FSE 2006, Graz, Austria, March 15-17, 2006, Revised Selected Papers*, volume 4047 of *Lecture Notes in Computer Science*, pages 310–327. Springer, 2006. doi:[10.1007/11799313_20](https://doi.org/10.1007/11799313_20).
- [JN22] Ashwin Jha and Mridul Nandi. A Survey on Applications of H-Technique: Revisiting Security Analysis of PRP and PRF. *Entropy*, 24(4):462, 2022. doi:[10.3390/E24040462](https://doi.org/10.3390/E24040462).
- [KO62] Anatolii Karatsuba and Yu Ofman. Multiplication of Multidigit Numbers on Automata. *Soviet Physics Doklady*, 7:595, 12 1962.
- [KR11] Ted Krovetz and Phillip Rogaway. The Software Performance of Authenticated-Encryption Modes. In Antoine Joux, editor, *Fast Software Encryption — 18th International Workshop, FSE 2011, Lyngby, Denmark, February 13-16, 2011, Revised Selected Papers*, volume 6733 of *LNCS*, pages 306–327. Springer, 2011. doi:[10.1007/978-3-642-21702-9_18](https://doi.org/10.1007/978-3-642-21702-9_18).
- [Kum18] Manish Kumar. Security of XCB and HCTR. *MA thesis. Indian Statistical Institute*, July 2018. Available from arXiv as paper 2308.06082: <https://doi.org/10.48550/arXiv.2308.06082>.
- [Lee24] Byeonghak Lee. A BBB Secure Accordion Mode from HCTR, June 2024. Available from: <https://csrc.nist.gov/csrc/media/Presentations/2024/a-bbb-secure-accordion-mode-from-hctr/images-media/sess-8-lee-acm-workshop-2024.pdf>.
- [LGR21] Julia Len, Paul Grubbs, and Thomas Ristenpart. Partitioning Oracle Attacks. In Michael D. Bailey and Rachel Greenstadt, editors, *30th USENIX Security Symposium, USENIX Security 2021, August 11-13, 2021*, pages 195–212. USENIX Association, 2021. Available from: <https://www.usenix.org/conference/usenixsecurity21/presentation/len>.
- [Lié23] Jean Liénardy. Padding-based forgeries in the mode XOCB. *IACR Cryptol. ePrint Arch.*, page 637, 2023.
- [Men15] Bart Mennink. Optimally Secure Tweakable Blockciphers. *Cryptology ePrint Archive*, Paper 2015/363, 2015. Available from: <https://eprint.iacr.org/2015/363>.

- [Mer79] Ralph Charles Merkle. *Secrecy, authentication, and public key systems*. PhD thesis, Department of Electrical Engineering, Stanford University, Stanford, 1979.
- [Mer89] Ralph Charles Merkle. One Way Hash Functions and DES. In Gilles Brassard, editor, *Advances in Cryptology - CRYPTO '89, 9th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 1989, Proceedings*, volume 435 of *Lecture Notes in Computer Science*, pages 428–446. Springer, 1989. doi:10.1007/0-387-34805-0_40.
- [MF04] David A. McGrew and Scott R. Fluhrer. The Extended Codebook (XCB) Mode of Operation. *IACR Cryptol. ePrint Arch.*, page 278, 2004. Available from: <http://eprint.iacr.org/2004/278>.
- [MF07] David A. McGrew and Scott R. Fluhrer. The Security of the Extended Codebook (XCB) Mode of Operation. In Carlisle M. Adams, Ali Miri, and Michael J. Wiener, editors, *Selected Areas in Cryptography, 14th International Workshop, SAC 2007, Ottawa, Canada, August 16-17, 2007, Revised Selected Papers*, volume 4876 of *Lecture Notes in Computer Science*, pages 311–327. Springer, 2007. doi:10.1007/978-3-540-77360-3_20.
- [MI11] Kazuhiko Minematsu and Tetsu Iwata. Building Blockcipher from Tweakable Blockcipher: Extending FSE 2009 Proposal. In Liqun Chen, editor, *Cryptography and Coding - 13th IMA International Conference, IMACC 2011, Oxford, UK, December 12-15, 2011. Proceedings*, volume 7089 of *Lecture Notes in Computer Science*, pages 391–412. Springer, 2011. doi:10.1007/978-3-642-25516-8_24.
- [MM07] Kazuhiko Minematsu and Toshiyasu Matsushima. Tweakable Enciphering Schemes from Hash-Sum-Expansion. In K. Srinathan, C. Pandu Rangan, and Moti Yung, editors, *Progress in Cryptology - INDOCRYPT 2007, 8th International Conference on Cryptology in India, Chennai, India, December 9-13, 2007, Proceedings*, volume 4859 of *Lecture Notes in Computer Science*, pages 252–267. Springer, 2007. doi:10.1007/978-3-540-77026-8_19.
- [MMO85] Stephen M. Matyas, Carl H. Meyer, and Jonathan Oseas. Generating strong one-way functions with cryptographic algorithms. *IBM Technical Disclosure Bulletin*, 27:5658–5659, 1985.
- [Mon85] Peter L. Montgomery. Modular Multiplication without Trial Division. *Mathematics of Computation*, 44(170):519–521, 1985.
- [MT05] Kazuhiko Minematsu and Yukiyasu Tsunoo. Hybrid Symmetric Encryption Using Known-Plaintext Attack-Secure Components. In Dongho Won and Seungjoo Kim, editors, *Information Security and Cryptology - ICISC 2005, 8th International Conference, Seoul, Korea, December 1-2, 2005, Revised Selected Papers*, volume 3935 of *Lecture Notes in Computer Science*, pages 242–260. Springer, 2005. doi:10.1007/11734727_21.
- [MV04] David A. McGrew and John Viega. Arbitrary block length mode, April 2004. Available from: <https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=6ff1a8bfcd7a214e1cf2c9c501e724ef753937e1>.
- [NIS15a] NIST. FIPS PUB 180-4 — Secure Hash Standard. *Federal information processing standards (FIPS)*, National Institute of Standards and Technology, Gaithersburg, MD, August 2015.

- [NIS15b] NIST. FIPS PUB 202 — SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions. *Federal information processing standards (FIPS)*, National Institute of Standards and Technology, Gaithersburg, MD, August 2015.
- [NR99] Moni Naor and Omer Reingold. On the Construction of Pseudorandom Permutations: Luby-Rackoff Revisited. *J. Cryptol.*, 12(1):29–66, 1999. doi:10.1007/PL00003817.
- [NRS14] Chanathip Namprempre, Phillip Rogaway, and Thomas Shrimpton. Reconsidering Generic Composition. In Phong Q. Nguyen and Elisabeth Oswald, editors, *Advances in Cryptology - EUROCRYPT 2014 - 33rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Copenhagen, Denmark, May 11-15, 2014. Proceedings*, volume 8441 of *Lecture Notes in Computer Science*, pages 257–274. Springer, 2014. doi:10.1007/978-3-642-55220-5_15.
- [NSS24] Yusuke Naito, Yu Sasaki, and Takeshi Sugawara. Committing Wide Encryption Mode with Minimum Ciphertext Expansion. *IACR Cryptol. ePrint Arch.*, page 1257, 2024. Available from: <https://eprint.iacr.org/2024/1257>.
- [OANWO] Jack O’Connor, Jean-Philippe Aumasson, Samuel Neves, and Zooko Wilcox-O’Hearn. The BLAKE3 cryptographic hash function. Available from: <https://github.com/BLAKE3-team>.
- [Pat08] Jacques Patarin. The “Coefficients H” Technique. In Roberto Maria Avanzi, Liam Keliher, and Francesco Sica, editors, *Selected Areas in Cryptography, 15th International Workshop, SAC 2008, Sackville, New Brunswick, Canada, August 14-15, Revised Selected Papers*, volume 5381 of *Lecture Notes in Computer Science*, pages 328–345. Springer, 2008. doi:10.1007/978-3-642-04159-4_21.
- [RBH17] Sondre Rønjom, Navid Ghaedi Bardeh, and Tor Helleseeth. Yoyo Tricks with AES. In Tsuyoshi Takagi and Thomas Peyrin, editors, *ASIACRYPT I*, volume 10624 of *Lecture Notes in Computer Science*, pages 217–243. Springer, 2017. doi:10.1007/978-3-319-70694-8_8.
- [Rog04] Phillip Rogaway. Efficient Instantiations of Tweakable Blockciphers and Refinements to Modes OCB and PMAC. In *Advances in Cryptology — ASIACRYPT 2004, 10th International Conference on the Theory and Application of Cryptology and Information Security, Jeju Island, Korea, December 5-9, 2004, Proceedings*, pages 16–31, 2004. doi:10.1007/978-3-540-30539-2_2.
- [RW72] Michael O. Rabin and Shmuel Winograd. Fast evaluation of polynomials by rational preparation. *Communications on Pure and Applied Mathematics*, 25(4):433–458, 1972. Available from: <https://onlinelibrary.wiley.com/doi/abs/10.1002/cpa.3160250405>, arXiv:<https://onlinelibrary.wiley.com/doi/pdf/10.1002/cpa.3160250405>, doi:<https://doi.org/10.1002/cpa.3160250405>.
- [Sar09] Palash Sarkar. Efficient tweakable enciphering schemes from (block-wise) universal hash functions. *IEEE Trans. Inf. Theory*, 55(10):4749–4760, 2009. doi:10.1109/TIT.2009.2027487.

- [Sar11] Palash Sarkar. Tweakable enciphering schemes using only the encryption function of a block cipher. *Inf. Process. Lett.*, 111(19):945–955, 2011. doi:10.1016/J.IPL.2011.06.014.
- [ST13] Thomas Shrimpton and R. Seth Terashima. A Modular Framework for Building Variable-Input-Length Tweakable Ciphers. In Kazue Sako and Palash Sarkar, editors, *Advances in Cryptology — ASIACRYPT 2013 — 19th International Conference on the Theory and Application of Cryptology and Information Security, Bengaluru, India, December 1-5, 2013, Proceedings, Part I*, volume 8269 of *LNCS*, pages 405–423. Springer, 2013. doi:10.1007/978-3-642-42033-7_21.
- [TSS⁺24] Atsushi Tanaka, Rentaro Shiba, Kosei Sakamoto, Mostafizar Rahman, Takuro Shiraya, and Takanori Isobe. ASURA: An Efficient Large-State Tweakable Block Cipher for ARM Environment. In Sourav Mukhopadhyay and Pantelimon Stanica, editors, *Progress in Cryptology - INDOCRYPT 2024 - 25th International Conference on Cryptology in India, Chennai, India, December 18-21, 2024, Proceedings, Part I*, volume 15495 of *Lecture Notes in Computer Science*, pages 143–164. Springer, 2024. doi:10.1007/978-3-031-80308-6_7.
- [WFW05] Peng Wang, Dengguo Feng, and Wenling Wu. HCTR: A Variable-Input-Length Enciphering Mode. In Dengguo Feng, Dongdai Lin, and Moti Yung, editors, *Information Security and Cryptology, First SKLOIS Conference, CISC 2005, Beijing, China, December 15-17, 2005, Proceedings*, volume 3822 of *Lecture Notes in Computer Science*, pages 175–188. Springer, 2005. doi:10.1007/11599548_15.
- [WMX⁺24] Peng Wang, Shuping Mao, Ruozhou Xu, Jiwu Jing, and Yuewu Wang. How to Recover the Full Plaintext of XCB. Cryptology ePrint Archive, Paper 2024/1527, 2024. Available from: <https://eprint.iacr.org/2024/1527>.
- [Zha22] Ping Zhang. Universal tweakable Even-Mansour cipher and its applications. *Frontiers of Computer Science*, 17(4):2095–2236, 2022. doi:10.1007/s11704-022-1466-1.