IACR Transactions on Cryptographic Hardware and Embedded Systems
ISSN 2569-2925, Vol. 2025, No. 3, pp. 437–469.
DOI:https://doi.org/10.46586/tches.v2025.i3.437-469

Cymric: Short-tailed but Mighty

Beyond-birthday-bound Secure Authenticated Encryption for Short Inputs

Alexandre Adomnicăi[1], Wonseok Choi[2] †, Yeongmin Lee[3], Kazuhiko Minematsu[4] and Yusuke Naito[5]

[1] Independent Researcher, Paris, France, alexandre@adomnicai.me
[2] DGIST, Daegu, Korea, wonseok@dgist.ac.kr
[3] DESILO Inc., Seoul, Korea, yeongmin.lee@desilo.ai
[4] NEC, Kawasaki, Japan, k-minematsu@nec.com
[5] Mitsubishi Electric Corporation, Kanagawa, Japan, Naito.Yusuke@ce.MitsubishiElectric.co.jp

Abstract. Authenticated encryption (AE) is a fundamental tool in today's secure communication. Numerous designs have been proposed, including well-known standards such as GCM. While their performance for long inputs is excellent, that for short inputs is often problematic due to high overhead in computation, showing a gap between the real need for IoT-like protocols where packets are often very short. Existing dedicated short-input AEs are very scarce, the classical Encode-then-encipher (Bellare and Rogaway, Asiacrypt 2000) and Manx (Adomnicăi et al., CT-RSA 2023), using up to two block cipher calls. They have superior performance for (very) short inputs, however, security is up to $n/2$ bits, where n is the block size of the underlying block cipher. This paper proposes a new family of short-input AEs, dubbed Cymric, which ensures beyond-birthday-bound (BBB) security. It supports a wider range of input space than EtE and Manx with the help of one additional block cipher call (thus three calls). In terms of the number of block cipher calls, Cymric is the known minimum construction of BBB-secure AEs, and we also prove this is indeed minimal by presenting an impossibility result on BBB-secure AE with two calls. Finally, we show a comprehensive benchmark on microcontrollers to show performance advantage over existing schemes.

Keywords: Authenticated Encryption · Short input · Beyond birthday bound security

1 Introduction

Authenticated encryption (AE) is a symmetric-key encryption function that provides both confidentiality and integrity of messages. AE has been extensively used as a core component of security protocols, such as TLS, IPsec and SSH. As a general-purpose scheme, the standard AE schemes such as GCM [nis07a] and OCB [KR11] have been designed so that they can take variable length input, for both short and sufficiently long cases. For instance, GCM (using AES) accepts a message of length 0 (*i.e.*, empty string) to about $2^{32} \cdot 128$ bits ≈ 68 gigabyte. After the introduction of AE [BN00, KY01, Rog02], numerous AE proposals have followed this principle. The CAESAR competition and the

†This work was conducted while Wonseok was at Purdue University and the Georgia Institute of Technology.

Licensed under Creative Commons License CC-BY 4.0.

NIST Lightweight cryptography (LwC) project require their AE proposals to follow this principle as well.

However, real world applications often put a high priority on performance for short inputs. Typical examples are found in low-power wireless communication because of (*e.g.*) limited packet length from power constraints. For example, Sigfox limits packet lengths to a maximum of 12 bytes [sig17], EnOcean limits packet lengths to 9 or 14 bytes [eno20], and Bluetooth Low Energy (v4.0) supports payloads up to 33 bytes. Electronic Product Code (EPC) specified for RFIDs has just a 12-byte payload. Micro QR code can contain up to 15 bytes [qrc24]. For healthcare applications using tiny medical sensors, Narrow-Band IoT standards work with 1 to 4-byte payloads [MAMK18]. Andreeva et al. [ALP+19b] present more examples. A general-purpose AE scheme could be used for these applications, however, their performance on short inputs is not always satisfactory. Even though their long-input performance is excellent, short-input performance suffers due to computational overhead.

The problem of AE performance for short inputs has received attention from the research community in recent years. Iwata et al. [IMGM15] proposed an AE mode to reduce computational overhead for short inputs. A number of NIST LwC proposals, including the winner, Ascon [DEMS19], the finalists ForkAE [ALP+19a] and Romulus [GIK+19], claimed good performance on short inputs. However, all of these schemes are general-purpose, and therefore also supporting sufficiently long inputs.

An interesting alternative direction has been proposed by Adomnicăi, Minematsu, and Shikata [AMS23] at CT-RSA 2023. They pointed out that the known general-purpose AEs built on block ciphers need at least three block cipher calls. Encode-then-Encipher (EtE) [BR00] is the most primitive AE mode that needs only one call. However, the total input length is at most one n-bit block when using an n-bit block cipher. This observation raised the question of what could be achieved with *two* calls. As an answer, they proposed Manx, a family of AE schemes dedicated to short inputs. It uses two calls and covers a larger area than EtE, *e.g.*, a plaintext of about n bits. Concretely, for ν-bit nonce, α-bit associated data, and a message of ℓ bits, Manx supports $\nu = n/2$ and $\ell = n$ with tiny α and ensures $n/2$-bit security[1]. Their microcontroller benchmarks demonstrated significant performance improvements over widely used modes.

Our contributions. The work of [AMS23] fills the gap between EtE and (say) GCM and OCB, and this further inspires questions. One such question is: *what could be done by three block cipher calls*? At first glance, the answer was already given, as COFB [CIMN17, BCI+19] encrypts an n-bit plaintext using three calls. However, the security of COFB is about $n/2$ (more precisely, $n/2 - \log_2 n$) bits, and since we do not support long inputs, it is natural to ask if we could achieve stronger security than $n/2$ bits, *i.e.*, beyond-birthday-bound (BBB) security, with the same number of calls. The birthday bound limit is critical for block ciphers of a small block size. Even with AES or other 128-bit block ciphers, this may not pose a threat in the distant future. For instance, Microsoft and Amazon expressed their concerns about the continued use of GCM with a 128-bit cipher (in the general context) in the comment to the review process of FIPS 197 (specifying AES) by NIST[2] since 64-bit security with respect to data complexity may not be enough in the near future (although their concerns are specific to their applications, such as cloud computing). Note that the existing short-input AEs (EtE, Manx) are also only $n/2$-bit secure at best. Therefore, our goal is to explore the smallest AE construction that achieves BBB security.

We provide a constructive solution by presenting Cymric, a family of AE modes tailored to short inputs that use three calls and achieve BBB security. Concretely, the first member,

[1] More precisely, a member of Manx (Manx2) requires $2\nu + \alpha + \ell < 2n$ and ensures $\min\{\nu, n/2\}$-bit security.

[2] https://csrc.nist.gov/News/2022/proposal-to-revise-sp-800-38a

Cymric1, operates over the input space (roughly) defined by $\nu + \alpha \leq n$ and $\nu + \ell \leq n$, ensuring full n-bit security. The second member, Cymric2, covers the input area (roughly) defined by $\nu + \alpha \leq n$ and $\ell \leq n$, ensuring $2n/3$-bit security. The covered input areas are significantly expanded from EtE/Manx, thereby enhancing applicability. Moreover, similar to many general-purpose AEs including GCM, the total output length of Cymric's encryption is $\nu + \alpha + \ell + t$ for a t-bit tag, hence it does not have unnecessary output expansion. This is a sharp difference from EtE and Manx as they have n- or $2n$-bit output regardless of message length, and is highly beneficial for IoT-like protocols where communication bandwidth is very limited. In addition, unlike EtE and Manx, our schemes do not require the block cipher inverse operation, also known as inverse-free, which reduces implementation size and sometimes boosts speed (AES's inverse is often slower than the forward on constrained devices, see *e.g.*, [GL12]). We also show that, using at most two n-bit block cipher calls with linear operations, it is impossible to build an AE supporting n-bit message and ensuring BBB security, thereby indicating Cymric2's optimality. We remark that our framework follows existing studies on showing impossibility/optimality of designs using a given number of primitive calls (see Sect. 6.1). Table 1 shows a comparison between Cymric and other popular AE schemes and dedicated short-input schemes.

The core idea behind Cymric lies in recent progress in security proofs of EWCDM (Encrypted Wegman-Carter with Davies-Meyer) and its variants [CLL24] , backed by the advanced/refined version of Mirror theory [CLLL21, CDN+23]. We also introduce a special *intermediate* world to prove the bound for Cymric2, which is a novel technique that may be of independent interest. All of these efforts enable us to prove BBB security, including the full-bit security for Cymric1. Although the structures of the Sum-of-Permutation PRF SoP [Luc00, DHT17] and EWCDM [CS16] are found within Cymric, Cymric is not a black-box composition of them. Therefore, our proofs are not directly based on the state-of-the-art provable security results for them [DHT17, DNS20, CLL24]. The structure of Cymric reduces the number of block cipher calls while inheriting the strong security of SoP and EWCDM.

Potential variants using advanced primitives. Cymric requires three block cipher calls, and each block's inputs (or the key) are distinct from those of the other blocks to maintain security guarantees. This separation is achieved through domain separation and the use of independent keys, which slightly reduces its supported input size and requires additional memory. This limitation can be addressed by using tweakable block ciphers (TBCs), *i.e.*, by including domain separation bits in the tweak inputs, which also expands the message space of Cymric by 2 bits. A few TBC candidates with small tweak spaces are designed specifically for such applications, *e.g.*, TweGIFT or TweAES [CDJ+21]. Designing efficient AEs for short inputs based on larger tweak-size TBCs or forkciphers [ALP+19b] could also be an interesting open problem. In such cases, the tweak space could be utilized to absorb associated data.

Potential use cases. Cymric could be used for applications/protocols with short inputs, as was mentioned earlier. Its BBB security could significantly enhance security, especially in terms of data complexity, making it highly relevant for many IoT use cases. This is particularly true in scenarios where rekeying is difficult (*e.g.* hardwired systems) and the device could have a long lifetime, such as when powered by passive energy. Here are some examples:

1. Some protocols, such as Voice over Internet Protocol (VoIP) or the forthcoming Short-packet communication (SPC) in 5G, must be able to handle small data packets in a high-throughput/low-latency network. In the latter case, the payload size can be a few bytes [BJL+14, BPN+16]. One concrete example of SPC is massive machine-type communication (mMTC), which targets factory automation or drone control.

This area has been actively studied in the network research community, focusing on payload sizes around 100 bits [DKP16, DLS18].

2. CAN (Controller Area Network) was developed for control systems in cars but is also widely adopted in industrial networks consisting of embedded processors [can24]. The lack of cryptographic protection in industrial networks is widely recognized as a serious security concern [Dav24]. CAN has a data rate of 1 Mbit/sec and a payload of 8 bytes. Suppose a group of 16 network controllers in a factory shares a secret key for protected communication using an AE scheme, and the security threshold[3] is 2^{-48} (*i.e.*, we want to ensure that the success probability of any adversary who observes the network is at most this value). A typical birthday bound of an AE scheme (*e.g.*, GCM or OCB) is $\sigma^2/2^n$, where σ denotes the total communicated data in blocks, and n denotes the block size. Suppose we use AES ($n = 128$). Each controller unit can send around $2^{29.33}$ blocks per day and the total communicated data can reach the threshold ($2^{(128-48)/2}$) after 102 days, since $2^{(128-48)/2-(29.33+\log_2 16)} \approx 101.8$. If the group consists of 32 controllers, the limit reduces to 51 days. The use of a BBB-secure AE will greatly relax these limits.

3. The Constrained Application Protocol (CoAP) [SHB14] is a lightweight protocol designed for constrained devices and networks processing short payloads, sometimes as small as a single byte. OSCORE (Object Security for COnstrained Restful Environments) [SMPS19] enhances CoAP by providing encryption, integrity, and replay protection directly at the application layer. In essence, OSCORE transforms an unprotected CoAP message into a protected one. By securing only the parts of the CoAP message that require confidentiality or integrity, OSCORE handles minimal payloads using AES-CCM, with as little as 3 bytes of additional data (1 byte for versioning and 2 bytes for algorithm identifiers) and a 1-byte payload. Note that OSCORE relies on a pre-established security context, which includes cryptographic material such as keys and counters shared between communicating devices. Setting up this security context is a non-trivial process that typically involves a key exchange protocol and requires careful management to maintain synchronization between devices. Because updating the security material may be necessary for various reasons (*e.g.*, approaching the key usage limits [HT24]), having a BBB-secure AE instead of AES-CCM would be beneficial as it would contribute to lower the update frequency.

We also note that the NIST LwC winner Ascon [DEMS19, TMC+24] has a 256-bit capacity, defined as the width of the underlying permutation minus the rate, where the rate is the number of input bits. This results in an online security level of 128 bits. This suggests the relevance of BBB security for 128-bit block ciphers, even in lightweight applications. In general, BBB-secure AEs typically have complex structures, resulting in significant computational overhead and poor performance for short inputs, as exemplified by XOCB [BHI+23]. When message lengths vary in the target application, we can improve average efficiency while maintaining BBB security by processing short messages with Cymric and long messages with a general-purpose BBB-secure AE, with a proper key separation. Short messages can happen *e.g.*, when encrypting a transcript of handshake in TLS or a simple chat reply in messaging protocols.

We stress that the utility of Cymric for the use cases suggested above will depend on many factors determined by the application. In SPC in 5G (the first example), the underlying platforms are unlikely to be constrained for attaining low-latency communication, hence the computational merit of Cymric over other BBB-secure AEs may be less visible. For the second example, if the CAN network has a lower data rate, it makes the security limit less critical, and a conventional up-to-birthday-secure AE would suffice. Moreover,

[3] For example, TLS 1.3 aims for security threshold of 2^{-57} to 2^{-60} [GTW24].

measuring the real performance gain will not be easy as it depends on (*e.g.*) message distributions, platforms, and network environment. If we want to measure the performance of Cymric in OSCORE (the third example) running in the real environment, we would need an intensive engineering study such as Guunarsson et al. [GMHT22], which is beyond our scope. Nevertheless, the inherent computation gain of Cymric over CCM (OSCORE default) is around 25% since CCM needs 4 block cipher calls for one block, and we achieve stronger security.

The value of Cymric could also be discussed from more general viewpoints; if the performance of AE is a bottleneck, an obvious option is to upgrade the computing platform. On the other hand, the reductions in time and memory matter, as it shows that Cymric can outperform the state of the art when instantiated with ciphers available off-the-shelf (and the improvement factor could scale with other ciphers and/or countermeasures against physical attacks). If a relevant use case requires support for long messages as well (*e.g.*, for software updates), the same cipher inside Cymric could also be used with another mode supporting long messages. Although this would increase the code size, Cymric's extreme simplicity means that most of the memory footprint comes from the cipher itself. Upgrading the computing platform may be a viable solution if encryption performance is a bottleneck, but we want to stress that challenges beyond cryptographic components may arise. As an example, modern cars contain more than a hundred microcontrollers, ranging from 8- to 32-bit: such upgrades could lead to engineering challenges and incur non-negligible costs. Consequently, despite a large benchmark we provide (see below), there is an inherent limitation in evaluating the utility of our proposal in the current and future applications. We leave the evaluation of Cymric for these potential applications in the real environment as an interesting future work.

Benchmark. We report software performance results on two microprocessor architectures: 8-bit AVR and 32-bit ARM Cortex-M4, using two benchmarks. The first benchmark compares Cymric to other commonly used modes, carefully selected for their efficiency over short inputs, by instantiating them with AES-128 as the underlying block cipher. The second benchmark evaluates Cymric against Ascon, the upcoming lightweight cryptography standard, and other NIST LwC finalists. For this, we instantiate Cymric with two lightweight block ciphers: GIFT-128 [BPP+17] introduced at CHES 2017 and used in multiple LwC candidates and LEA-128 [HLK+14], a Korean national standard [KSX16] and an ISO/IEC standard [ISO19] that is highly efficient on embedded platforms and has undergone extensive cryptanalysis. Both benchmarks consider multiple payload sizes to provide a comprehensive assessment across various use cases. On both platforms and for both benchmarks, Cymric instantiations demonstrate comparable or superior efficiency to all other AEs considered, including Ascon. Overall, its straightforward design facilitates easy deployment, minimizing the risk of implementation failures.

2 Preliminaries

Notation. For integers i and j with $i < j$, let $[i..j]$ denote $\{i, i+1, i+2, \ldots, j\}$. Let $\{0,1\}^*$ be the set of all bit strings, including the empty string ε. For any set $\mathcal{X}$, let $\mathcal{X}^{\leq i} := \bigcup_{j \in [0..i]} \mathcal{X}^j$. For $X \in \{0,1\}^*$, $|X|$ denotes its bit length, where $|\varepsilon| = 0$. A concatenation of bit strings X and Y is denoted as $X \,\|\, Y$, or XY, if it is clear from the context. The parsing of X into n-bit blocks for a positive integer n is denoted by

$$(X[1], X[2], \ldots, X[m]) \xleftarrow{n} X,$$

where $m = |X|_n := \lceil |X|/n \rceil$, $X[1] \,\|\, X[2] \,\|\, \ldots \,\|\, X[m] = X$, $|X[i]| = n$ for $1 \leq i < m$ and $0 < |X[m]| \leq n$ when $|X| > 0$. When $X = \varepsilon$, we let $X[1] \xleftarrow{n} X$ with $X[1] = \varepsilon$.

Table 1: Comparison of AE schemes based on an n-bit block cipher. MUL denotes a multiplication over GF(2^n). Max message length denotes the maximum bit length of plaintext/ciphertext allowed, where ν and α denote the bit lengths of nonce and associated data. Min Calls denotes the minimum number of block cipher calls required for non-empty messages. Expansion indicates additional blowup in bandwidth (See Sect. 2.1).

Scheme	Max. message length	Primitive	Min. calls	Security	Expansion	Ref
OCB	any	SPRP	4	$n/2$	No	[KR11]
GCM	any	PRP, MUL	$3^{\dagger 1}$	$n/2$	No	[nis07a, NOMI15]
CCM	any	PRP	4	$n/2$	No	[nis07b]
XOCB	any	SPRP	9	$2n/3$	No	[BHI+23]
EtE	$n-\nu-\alpha$	SPRP	1	$n/2$	Yes	[BR00]
Manx2	n	SPRP	2	$n/2^{\dagger 2}$	Yes	[AMS23]
Cymric1	$n-\nu$	PRP	3	n	No	This work
Cymric2	n	PRP	3	$2n/3$	No	This work

†1: additional $\mathsf{GF}(2^n)$ multiplications (two when $\nu = 96$ and four otherwise)
†2: optimal value achieved when nonce is $n/2$ bits

The set of all sequences that consist of b pairwise distinct elements of $\mathcal{X}$ is denoted $\mathcal{X}^{*b}$. For integers $1 < b < a$, we will write $(a)_b = a(a-1)\cdots(a-b+1)$ and $(a)_0 = 1$ by convention. If $|\mathcal{X}| = a$, then $(a)_b$ becomes the size of $\mathcal{X}^{*b}$.

Let 0^i be the string of i zero bits. We write 10^i for $1 \,\|\, 0^i$. When $|X| \geq i$, $\mathtt{msb}_i(X)$ denotes the first (left) i bits of X. Similarly, $\mathsf{lsb}_i(X)$ denotes the last (right) i bits of X. For $X \in \{0,1\}^{\leq n}$, $\mathtt{pad}_n(X) = X$ when $|X| = n$ and $\mathtt{pad}_n(X) = X \,\|\, 10^{n-|X|-1}$ otherwise. This is also known as "one-zero padding". Note that $\mathtt{pad}_n(\cdot)$ is not injective.

Throughout this paper, we fix positive integers n to denote the block size. Given a non-empty finite set $\mathcal{X}$, $x \leftarrow_\$ \mathcal{X}$ denotes that x is chosen uniformly at random from $\mathcal{X}$. $|\mathcal{X}|$ means the number of elements in $\mathcal{X}$. The set of all permutations of $\{0,1\}^n$ is simply denoted $\mathsf{Perm}(n)$. For some positive integer m, the set of all functions with domain $\{0,1\}^n$ and codomain $\{0,1\}^m$ is simply denoted by $\mathsf{Func}(n,m)$. For a keyed function $F : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ with key space $\mathcal{K}$ and non-empty sets $\mathcal{X}$ and $\mathcal{Y}$, we will denote $F(K,\cdot)$ by $F_K(\cdot)$ for $K \in \mathcal{K}$. An n-bit random permutation P is uniformly distributed over $\mathsf{Perm}(n)$. An n-bit-to-m-bit random function F is uniformly distributed over $\mathsf{Func}(n,m)$.

Block Cipher and (S)PRP security. Let $E : \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ be an n-bit block cipher with key space $\mathcal{K}$. Its inverse is denoted by $E^{-1} : \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ so that $E_K^{-1}(E_K(M)) = M$ for any $(K,M) \in \mathcal{K} \times \{0,1\}^n$. We will consider a distinguisher $\mathcal{A}$ that makes oracle queries to E and E^{-1}, and returns a single bit (0 or 1). The advantage of $\mathcal{A}$ in breaking the SPRP and PRP security of E are defined as

Definition 1.

$$\mathbf{Adv}^{\mathsf{sprp}}_E(\mathcal{A}) = \left|\Pr\left[K \leftarrow_\$ \mathcal{K} : \mathcal{A}^{E_K, E_K^{-1}} = 1\right] - \Pr\left[\mathsf{P} \leftarrow_\$ \mathsf{Perm}(n) : \mathcal{A}^{\mathsf{P},\mathsf{P}^{-1}} = 1\right]\right|,$$
$$\mathbf{Adv}^{\mathsf{prp}}_E(\mathcal{A}) = \left|\Pr\left[K \leftarrow_\$ \mathcal{K} : \mathcal{A}^{\mathsf{E}_K} = 1\right] - \Pr\left[\mathsf{P} \leftarrow_\$ \mathsf{Perm}(n) : \mathcal{A}^{\mathsf{P}} = 1\right]\right|,$$

A (q, tm)-adversary against E is an algorithm that makes at most q queries to E_K or E_K^{-1} and runs in time at most tm. When considering an information-theoretic adversary, i.e., without time restriction, we will drop the parameter tm.

2.1 Nonce-based Authenticated Encryption

A nonce-based authenticated encryption (nAE) scheme is a tuple $\Pi = (\mathsf{Enc}, \mathsf{Dec})$, where Enc and Dec are called (authenticated) encryption and decryption algorithms, respectively.

The encryption algorithm Enc takes as input a key $K \in \mathcal{K}$, a nonce $N \in \mathcal{N}$, an associated data (also known as header) $A \in \mathcal{H}$, and a message $M \in \mathcal{M}$, and outputs a ciphertext $C \in \{0,1\}^*$ and a tag $T \in \mathcal{T}$. The decryption algorithm Dec takes as input a tuple $(K, N, A, C, T) \in \mathcal{K} \times \mathcal{N} \times \mathcal{H} \times \{0,1\}^* \times \mathcal{T}$, outputs either a message $M \in \mathcal{M}$ or a special symbol $\perp$ indicating the authentication failure. We require that

$$\mathsf{Dec}(K, N, A, C, T) = M, \text{ where } (C, T) = \mathsf{Enc}(K, N, A, M)$$

for any tuple $(K, N, A, M) \in \mathcal{K} \times \mathcal{N} \times \mathcal{H} \times \mathcal{M}$. The tag length t is fixed in advance. If $|C| = |M|$ holds for any valid queries, we say Π has no (bandwidth) expansion. We will write $\mathsf{Enc}_K(N, A, M)$ and $\mathsf{Dec}_K(N, A, C, T)$ to denote $\mathsf{Enc}(K, N, A, M)$ and $\mathsf{Dec}(K, N, A, C, T)$, respectively.

The goal of an adversary $\mathcal{A}$ against the nonce-based AE security of Π is to distinguish the real world $(\mathsf{Enc}_K, \mathsf{Dec}_K)$ using a secret key $K \leftarrow_\$ \mathcal{K}$ and the ideal world. The ideal world is a tuple $(\mathsf{Rand}, \mathsf{Rej})$, where Rand returns an independent random string of length $|\mathsf{Enc}_K(N, A, M)|$ and Rej always returns $\perp$ for every decryption query. We assume that $\mathcal{A}$ does not make a decryption query by reusing any previous encryption query. The advantage of $\mathcal{A}$ breaking the nAE-security of Π is defined as follows.

Definition 2.

$$\mathbf{Adv}^{\mathsf{nAE}}_{\Pi}(\mathcal{A}) = \left|\Pr[K \leftarrow_\$ \mathcal{K} : \mathcal{A}^{\mathsf{Enc}_K, \mathsf{Dec}_K} = 1] - \Pr[\mathcal{A}^{\mathsf{Rand}, \mathsf{Rej}} = 1]\right|.$$

A (q_e, q_d, tm)-adversary against Π is an algorithm that makes at most q_e nonce-respecting encryption queries to its first oracle and at most q_d decryption queries to its second oracle, and running in time at most tm. When we consider information-theoretic security, we will drop the parameter tm.

2.2 Coefficient-H Technique

We will use Patarin's coefficient-H technique [Pat90]. The goal of this technique is to upper bound the adversarial distinguishing advantage between a real construction and its ideal counterpart. In the ideal and the real worlds, an information-theoretic adversary $\mathcal{A}$ is allowed to make q queries to certain oracles (with the same oracle interfaces), denoted $\mathcal{S}_0$ and $\mathcal{S}_1$, respectively. The interaction between the adversary $\mathcal{A}$ and the oracle determines a "transcript" $\tau \in \Omega^q$; it contains all the information obtained by $\mathcal{A}$ during the interaction. We call a transcript τ *attainable* if the probability of obtaining τ in the ideal world is non-zero.

We partition the set of attainable transcripts Θ into a set of "good" transcripts Θ_{good} such that the probabilities of obtaining some transcript $\tau \in \Theta_{\mathsf{good}}$ are close in the real world and the ideal world, and a set Θ_{bad} of "bad" transcripts such that the probability of obtaining any $\tau \in \Theta_{\mathsf{bad}}$ is small in the ideal world. The coefficient-H technique is summarized in the following lemma.

Lemma 1. *Let $\Theta = \Theta_{\mathsf{good}} \sqcup \Theta_{\mathsf{bad}}$ be a partition of the set of attainable transcripts, where there exists a non-negative ϵ_1 such that for any $\tau \in \Theta_{\mathsf{good}}$, $\frac{\mathsf{p}^q_{\mathcal{S}_1}(\tau)}{\mathsf{p}^q_{\mathcal{S}_0}(\tau)} \geq 1 - \epsilon_1$, and there exists ϵ_2 such that $\sum_{\tau \in \Theta_{\mathsf{bad}}} \mathsf{p}^q_{\mathcal{S}_0}(\tau) \leq \epsilon_2$. Then, $\sum_{\tau \in \Theta} \max\left\{0, \mathsf{p}^q_{\mathcal{S}_0}(\tau) - \mathsf{p}^q_{\mathcal{S}_1}(\tau)\right\} \leq \epsilon_1 + \epsilon_2$.*

The proof could be found in *e.g.*, Chen and Steinberger [CS14].

3 Cymric: Amalgamating EWCDM and SoP

Existing Short-input AEs. As mentioned in the introduction, the lineup of known short-input AEs is limited. EtE [BR00] is the most well-known classic scheme, needing a single

n-bit block cipher call, and being able to encrypt (N, A, M), where $(|N|, |A|, |M|) = (\nu, \alpha, \ell)$, provided that $\nu + \alpha + \ell \leq n$. A slight extension of EtE based on permutation and hashing was proposed by Khovratovich [Kho14]. Manx [AMS23] is a family of two short-input AE modes, needing two calls. In particular, Manx2 works with the case $2\nu + \alpha + \ell \leq 2n$ and ensures $\min\{\nu, n/2\}$-bit security.

Our Proposals. We present Cymric, a family of short-input AEs with strong security. The core idea is to *amalgamating* EWCDM nonce-based MAC and SoP PRF, both providing beyond-birthday-bound security [CLL24, DHT17]. A generic composition could be used; however, Cymric minimizes computational effort by sharing the initial computations across both modes, reducing the total number of block cipher calls and achieving improved performance for short inputs.

As presented before, we use ν, α, ℓ, and t to denote $|N|$, $|A|$, $|M|$ ($|C|$), and $|T|$ of an encryption (decryption) query. We present two schemes. As in most popular modes, we fix ν and t in advance. Both are at most n bits, but α and ℓ may change depending on the query under certain restrictions involving ν.

Below we present our proposals. See also Fig. 1 for the pseudocodes and Fig. 2 for an illustration of their encryption routines.

3.1 Specification of Cymric1

The first member of Cymric is Cymric1. It supports an encryption query (N, A, M) such that $\nu + \alpha \leq n - 3$ and $\nu + \ell \leq n$. Note that this generally allows more flexible choice than EtE which requires $\nu + \alpha + \ell \leq n$. We use an n-bit block cipher E with two independent keys, K and K'. The encryption outputs (C, T) where $|C| = |M|$ and $|T| = t$. The encryption works as follows. Let $b \in \{0, 1\}$ is 0 when $\nu + \ell < n$ and 1 otherwise[4].

$$C = \mathtt{msb}_{|M|}\left(E_K(\mathtt{pad}_n(N \,\|\, A \,\|\, b0)) \oplus E_K(\mathtt{pad}_n(N \,\|\, A \,\|\, b1))\right) \oplus M.$$
$$T = \mathtt{msb}_t\left(E_{K'}(E_K(\mathtt{pad}_n(N \,\|\, A \,\|\, b0))) \oplus \mathtt{pad}_n(N \,\|\, M)\right),$$

where $b0$ and $b1$ are shorthand for $b \,\|\, 0$ and $b \,\|\, 1$. For a decryption query (N, A, C, T), Cymric1 first derives the candidate plaintext M, concatenates it with N and takes a sum with $E_K(\mathtt{pad}_n(N \,\|\, A \,\|\, b0))$ to derive a candidate tag. See Fig. 1 for the exact routine.

3.2 Specification of Cymric2

The second member, Cymric2, extends the input space of Cymric1, keeping three calls with a tiny change in the specification. For given ν-bit nonce N, Cymric2 accepts an associated data string A and a message M such that $\nu + \alpha \leq n - 3$ and $\ell \leq n$. Let $b \in \{0, 1\}$ be equal to 0 when $\ell = |M| < n$ and 1 otherwise. For encryption input (N, A, M), Cymric2 computes a tag (C, T) where $|C| = |M|$ and $|T| = t$ such that

$$C = \mathtt{msb}_{|M|}\left(E_K(\mathtt{pad}_n(N \,\|\, A \,\|\, b0)) \oplus E_K(\mathtt{pad}_n(N \,\|\, A \,\|\, b1))\right) \oplus M.$$
$$T = \mathtt{msb}_t\left(E_{K'}(E_K(\mathtt{pad}_n(N \,\|\, A \,\|\, b0))) \oplus \mathtt{pad}_n(M)\right).$$

The decryption is similarly performed as Cymric1. See Fig. 1.

3.3 Provable Security of Cymric

We present the security bounds of Cymric1 and Cymric2.

[4] By omitting b, we have a minor variant that allows $\nu + \ell \leq n - 1$ and $\nu + \alpha \leq n - 2$.

Algorithm Cymric1.$\mathcal{E}[E_K, E_{K'}](N, A, M)$

1 **if** $|M| < n - |N|$ **then** $b \leftarrow 0$
2 **else** $b \leftarrow 1$ $//|N| + |M| = n$
3 $Y[0] \leftarrow E_K(\mathtt{pad}_n(N \,\|\, A \,\|\, b0))$
4 $Y[1] \leftarrow E_K(\mathtt{pad}_n(N \,\|\, A \,\|\, b1))$
5 $C \leftarrow M \oplus \mathtt{msb}_{|M|}(Y[0] \oplus Y[1])$
6 $T \leftarrow Y[0] \oplus \mathtt{pad}_n(N \,\|\, M)$
7 $T \leftarrow \mathtt{msb}_t(E_{K'}(T))$
8 **return** (C, T)

Algorithm Cymric1.$\mathcal{D}[E_K, E_{K'}](N, A, C, T)$

1 **if** $|C| < n - |N|$ **then** $b \leftarrow 0$
2 **else** $b \leftarrow 1$ $//|N| + |C| = n$
3 $Y[0] \leftarrow E_K(\mathtt{pad}_n(N \,\|\, A \,\|\, b0))$
4 $Y[1] \leftarrow E_K(\mathtt{pad}_n(N \,\|\, A \,\|\, b1))$
5 $M \leftarrow C \oplus \mathtt{msb}_{|C|}(Y[0] \oplus Y[1])$
6 $\widehat{T} \leftarrow Y[0] \oplus \mathtt{pad}_n(N \,\|\, M)$
7 $\widehat{T} \leftarrow \mathtt{msb}_t(E_{K'}(\widehat{T}))$
8 **if** $\widehat{T} \neq T$ **return** $\perp$**, else return** M

Algorithm Cymric2.$\mathcal{E}[E_K, E_{K'}](N, A, M)$

1 **if** $|M| < n$ **then** $b \leftarrow 0$
2 **else** $b \leftarrow 1$ $//|M| = n$
3 $Y[0] \leftarrow E_K(\mathtt{pad}_n(N \,\|\, A \,\|\, b0))$
4 $Y[1] \leftarrow E_K(\mathtt{pad}_n(N \,\|\, A \,\|\, b1))$
5 $C \leftarrow M \oplus \mathtt{msb}_{|M|}(Y[0] \oplus Y[1])$
6 $T \leftarrow Y[0] \oplus \mathtt{pad}_n(M))$
7 $T \leftarrow \mathtt{msb}_t(E_{K'}(T))$
8 **return** (C, T)

Algorithm Cymric2.$\mathcal{D}[E_K, E_{K'}](N, A, C, T)$

1 **if** $|C| < n$ **then** $b \leftarrow 0$
2 **else** $b \leftarrow 1$ $//|C| = n$
3 $Y[0] \leftarrow E_K(\mathtt{pad}_n(N \,\|\, A \,\|\, b0))$
4 $Y[1] \leftarrow E_K(\mathtt{pad}_n(N \,\|\, A \,\|\, b1))$
5 $M \leftarrow C \oplus \mathtt{msb}_{|C|}(Y[0] \oplus Y[1])$
6 $\widehat{T} \leftarrow Y[0] \oplus \mathtt{pad}_n(M)$
7 $\widehat{T} \leftarrow \mathtt{msb}_t(E_{K'}(\widehat{T}))$
8 **if** $\widehat{T} \neq T$ **return** $\perp$**, else return** M

Figure 1: Algorithm of Cymric1 (top) and Cymric2 (bottom). For ν-bit nonce, α-bit AD, and ℓ-bit message, Cymric1 requires $\nu + \ell \leq n$ and Cymric2 requires $\nu \leq n$. For both, ν and t are fixed in advance. The bit length of AD, α, must follow $\nu + \alpha \leq n - 3$.

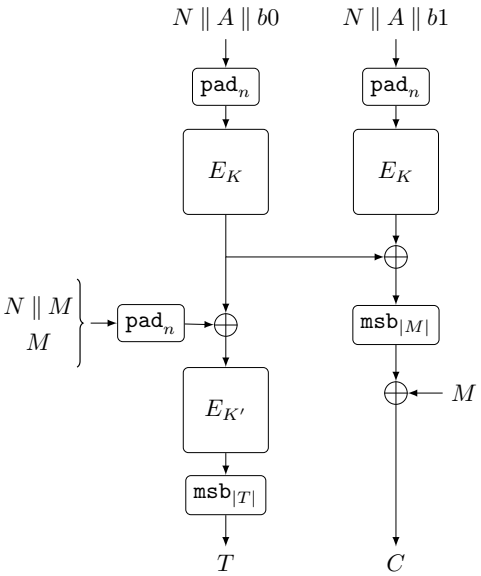

Figure 2: Encryption of Cymric. Cymric1 uses $N \,\|\, M$ for the middle XOR in the left branch whereas Cymric2 uses M.

Security bound of Cymric1**.**

Theorem 1. *Consider a* (q_e, q_d, tm)*-adversary against* Cymric1 *using an* n*-bit block cipher* $E : \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$*. If* $q_e \leq \frac{2^n}{48n^2}$*,* $q_d \leq 2^{t-1}$ *and* $n \geq 36$*, then we have*

$$\mathbf{Adv}^{\mathsf{nAE}}_{\mathsf{Cymric1}[E_K, E_{K'}]}(q_e, q_d, \mathsf{tm}) \leq 2\mathbf{Adv}^{\mathsf{prp}}_{E}(3q, \mathsf{tm} + 3qc) + \frac{8q}{2^n} + \frac{2q_d}{2^t},$$

where $q = q_e + q_d$ *and* c *is the time to compute* E *that depends on the computation models.*

The above bound tells that Cymric1 achieves approximately $(n-3)$-bit security by setting $t = n$.

Security bound of Cymric2**.**

Theorem 2. *Consider a* (q_e, q_d, tm)*-adversary against* Cymric2 *using an* n*-bit block cipher* $E : \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$*. We assume* $q_e + 2^{n-t} \cdot q_d \leq 2^{n-1}$ *and* $n \geq 36$*. If* $q_e \leq \frac{2^n}{48n^2}$*, then we have*

$$\mathbf{Adv}^{\mathsf{nAE}}_{\mathsf{Cymric2}[E_K, E_{K'}]}(q_e, q_d, \mathsf{tm}) \leq 2\mathbf{Adv}^{\mathsf{prp}}_{E}(3q, \mathsf{tm} + 3qc) + \frac{(12 + 2^{\frac{t}{2}})q_e}{2^n} + \frac{7q_d}{2^t},$$

where c *is the time to compute* E *that depends on the computation models.*

Corollary 1. *By letting* $t = 2n/3$ *and* $q = q_e + q_d$*, we have*

$$\mathbf{Adv}^{\mathsf{nAE}}_{\mathsf{Cymric2}[E_K, E_{K'}]}(q_e, q_d, \mathsf{tm}) \leq 2\mathbf{Adv}^{\mathsf{prp}}_{E}(3q, \mathsf{tm} + 3qc) + \frac{13q}{2^{2n/3}}.$$

The above bound tells that security of Cymric2 is $2n/3$ bits. Thus, Cymric2 extends the input space at the cost of quantitatively weaker (yet BBB) security than Cymric1.

4 Security Proofs

4.1 Proof of Theorem 1

Up to the PRP security of E, we replace E_K and $E^{-1}_{K'}$ into P_1 and P_2, respectively (note that the latter replacement is not for $E_{K'}$; this is for notational convenience). Precisely, the cost of this replacement is upper bounded by

$$2\mathbf{Adv}^{\mathsf{prp}}_{E}(3q, \mathsf{tm} + 3qc)$$

since $\mathcal{A}$ makes at most $3q$ block cipher queries.

Let $\mathsf{Enc}[P_1, P_2]$ denote the construction instantiated with permutations P_1 and P_2, and $\mathsf{Dec}[P_1, P_2]$ denotes the corresponding decryption oracle. Consider any adversary $\mathcal{A}$ interacts with either $\mathcal{S}_{\mathrm{real}} = (\mathsf{Enc}, \mathsf{Dec})$ or $\mathcal{S}_{\mathrm{ideal}} = (\$, \bot)$ where $\$$ denotes a truly random oracle with domain $\mathcal{N} \times \mathcal{H} \times \mathcal{M}$ that returns a tuple of a $|M|$-bit random string C and a t-bit random string T, and $\bot$ is an oracle that always returns 0.

To simplify the proof, we assume without loss of generality that (information-theoretic) $\mathcal{A}$ makes encryption queries first and then makes decryption queries. This is because we can construct $\mathcal{A}$ from any arbitrary $\mathcal{A}'$ as a subroutine as follows: $\mathcal{A}$ internally runs $\mathcal{A}'$. Once $\mathcal{A}'$ makes an encryption query, $\mathcal{A}$ forwards it to the oracle and forwards the answer from the oracle to $\mathcal{A}'$. If $\mathcal{A}'$ makes a decryption query, $\mathcal{A}$ records it and returns $\bot$ to $\mathcal{A}'$. After $\mathcal{A}'$ makes all queries, $\mathcal{A}'$ starts querying the recorded decryption queries to the oracle. This ensures the advantage of $\mathcal{A}'$ is less or equal than that of $\mathcal{A}$.

Let τ_e and τ_d denote the list of encryption queries and decryption queries, i.e.,

$$\tau_e = \{(N_1, A_1, M_1, C_1, T_1), \ldots, (N_{q_e}, A_{q_e}, M_{q_e}, C_{q_e}, T_{q_e})\},$$
$$\tau_d = \{(N_{q_e+1}, A_{q_e+1}, C_{q_e+1}, T_{q_e+1}, b_{q_e+1}), \ldots, (N_{q_e+q_d}, A_{q_e+q_d}, C_{q_e+q_d}, T_{q_e+q_d}, b_{q_e+q_d})\}.$$

Note that $\mathcal{A}$ always has $b_i = \bot$ for $i \in [q_e + 1..q_e + q_d]$ if $\mathcal{A}$ interacts with the ideal oracle.

Additional Information and Extended Transcripts. In the real world, at the end of the game, we will give *a message* M_i corresponding to each verification query-response pair $(N_i, A_i, C_i, T_i, b_i)$ for $i \in [q_e + 1..q_e + q_d]$. On the other hand, in the ideal world, messages M_i do not exist. Instead, the game will simulate those values to give them to an adversary. Note that the following simulation is done at the end of the interaction.

- For $i \in [1..q_e]$, first compute

$$\mathsf{msb}_{|M_i|}(S[N_i, A_i]) \coloneqq M_i \oplus C_i$$

  and $\mathsf{lsb}_{n-|M_i|}(S[N_i, A_i])$ is uniformly randomly sampled.

- For each $i \in [q_e+1..q_e+q_d]$, if $S[N_i, A_i]$ is undefined, assign a uniformly random value from $\{0,1\}^n$. After seeing $S[N_i, A_i]$, one also can compute $M_i = \mathsf{msb}_{|M_i|}(S[N_i, A_i]) \oplus C_i$.

Additionally, we provide the truncated bits of the tags corresponding to the encryption queries. More precisely, in the real world, we give the actual output of the construction, while in the ideal world, we give a uniformly random $(n-t)$-bit value for each encryption query. For simplicity, once this additional information is given, T_i for $i \in [1..q_e]$ is treated as an n-bit tag.

Let τ_a be the set of M_i for $i \in [q_e + 1..q_e + q_d]$. Let Θ be the set of all attainable transcripts in the ideal world and $\tau = (\tau_e, \tau_d, \tau_a) \in \Theta$.

Defining Intermediate Variables. This proof utilizes the extended Mirror theory stated in Theorem 5 and the coefficient-H technique stated in Lemma 1. For simplicity, we denote

$$N_{0,i} = N_i \parallel A_i \parallel b0,\ \ N_{1,i} = N_i \parallel A_i \parallel b1.$$

The core of the security proof is to estimate the number of possible ways of fixing evaluations P_1 and P_2 in a way that

$$P_1(N_{0,i}) \oplus P_1(N_{1,i}) = S_i \coloneqq S[N_i, A_i]$$

for $i \in [1..q_e + q_v]$, and

$$\begin{aligned} P_1(N_{0,i}) \oplus P_2(T_i) &= X_i \coloneqq \mathtt{pad}_n(N_i \parallel M_i) \text{ for } i \in [1..q_e], \\ P_1(N_{0,i}) \oplus P_2(T_i) &\neq X_i \coloneqq \mathtt{pad}_n(N_i \parallel M_i) \text{ for } i \in [q_e + 1..q_e + q_d]. \end{aligned}$$

We will identify

$$\begin{aligned} \mathcal{V}_1 &= \{P_1(N_{0,i})\} \cup \{P_1(N_{1,i})\}_{i \in [1..q_e+q_d]}, \\ \mathcal{V}_2 &= \{P_2(T_i)\}_{i \in [1..q_e]} \cup \left\{P_2(\langle 0 \rangle_{n-t} \parallel T_i), \ldots, P_2(\langle 2^{n-t}-1 \rangle_{n-t} \parallel T_i)\right\}_{i \in [q_e+1..q_e+q_d]} \end{aligned}$$

as sets of variables where $\langle j \rangle_{n-t}$ denotes $n-t$ bits representation of j for $0 \le j \le 2^{n-t}-1$. We also define $\mathcal{V} = \mathcal{V}_1 \sqcup \mathcal{V}_2$. Then we can construct the system of equations Γ_τ as defined in Section A. Let r be the number of decryption queries i such that $N_j \parallel A_j = N_i \parallel A_i$ for $j \le i$. We note that Γ_τ consists of $2q_e + q_d - r$ equations and $2^{n-t} \cdot q_d$ inequalities. To satisfy the conditions in Theorem 5, we must first define bad events on a transcript τ, and then we can apply the extended Mirror theory to each transcript that the bad event does not happen.

Defining and Bounding Bad Events. A transcript $\tau = (\tau_e, \tau_d, \tau_a)$ is defined as *bad* if one of the following conditions holds.

- $\mathsf{bad}_1 \Leftrightarrow$ there exists $(i_1, \ldots, i_n) \in [1..q_e]^{*n}$ such that $T_{i_1} = \cdots = T_{i_n}$.

- $\mathsf{bad}_2 \Leftrightarrow$ there exists $i \in [1..q]$ such that $S_i = 0^n$.
- $\mathsf{bad}_3 \Leftrightarrow$ there exists $(i, j) \in [1..q_e]^{*2}$ such that $T_i = T_j$ and
$$X_i \oplus X_j \in \{0^n, S_i, S_j, S_i \oplus S_j\}.$$

If a transcript τ is not bad, then it will be called a *good* transcript. The probability that the bad event occurs is obtained as follows:

- In the ideal world, for any $i \neq j$, we have $\Pr[T_i = T_j] = \frac{1}{2^n}$ for encryption queries. Therefore,
$$\Pr[\mathsf{bad}_1] \leq \frac{\binom{q_e}{n}}{(2^n)^{n-1}} \leq \left(\frac{q_e}{2^n}\right)^{n-1} \leq \frac{q_e}{2^n},$$
since $q_e \leq 2^{n-1}$.
- It is easy to see that
$$\Pr[\mathsf{bad}_2] \leq \frac{q}{2^n}.$$
- Fix $i \in [1..q_e]$. Since $N_i \neq N_j$ for all $j \in [1..q_e] \setminus \{i\}$, $X_i \oplus X_j$ cannot be 0^n. There exists at most one query j such that $N_j = N_i \oplus \mathsf{msb}_\nu(S_i)$. For such a query, the probability that $T_i = T_j$ is $\frac{1}{2^n}$. By combining the case of $S_i, S_j, S_i \oplus S_j$, we have
$$\Pr[\mathsf{bad}_3] \leq \frac{3q_e}{2^n}.$$

Therefore, we have

$$\Pr[\mathsf{bad}] \leq \Pr[\mathsf{bad}_1] + \Pr[\mathsf{bad}_2] + \Pr[\mathsf{bad}_3] \leq \frac{3q}{2^n} + \frac{2q_e}{2^n}. \tag{1}$$

Good Transcript Analysis. For a good transcript τ and its system Γ_τ, by assuming nonces are not repeated, we observe that

- Γ_τ is nice by $\neg(\mathsf{bad}_2 \wedge \mathsf{bad}_3)$;
- $\xi_{\max} \leq 2n - 1$ and $\xi_{\max}^2 n + \xi_{max} \leq 4n^3 \leq 2^{n/2}$ by $\neg\mathsf{bad}_1$.

Henceforth, we can apply recent result of Mirror theory [CLL24] and then we have

$$h(\Gamma_\tau) \geq \frac{(2^n - 2)_{|\mathcal{V}_1|}(2^n - 2)_{|\mathcal{V}_2|}}{2^{n(2q_e + q_d - r)}} \left(1 - \frac{2q_d}{2^t}\right).$$

For the completeness of the proof, we restate the Mirror theory in Theorem 5. In the ideal world, it simulates q_e tag values, q_e ciphertext blocks for encryption queries, and $(q_d - r)$ message blocks for decryption queries. Therefore, we see that

$$\mathsf{p}_{\mathcal{S}_0}^{q_e+q_d}(\tau) = \frac{1}{2^{n(2q_e+q_d-r)}},\ \mathsf{p}_{\mathcal{S}_1}^{q_e+q_d}(\tau) = \frac{h(\Gamma_\tau)}{(2^n)_{|\mathcal{V}_1|}(2^n)_{|\mathcal{V}_2|}}.$$

From the above, one has

$$\begin{aligned}
\frac{\mathsf{p}_{\mathcal{S}_1}^{q_e+q_d}(\tau)}{\mathsf{p}_{\mathcal{S}_0}^{q_e+q_d}(\tau)} &\geq \frac{(2^n - 2)_{|\mathcal{V}_1|}(2^n - 2)_{|\mathcal{V}_2|}}{(2^n)_{|\mathcal{V}_1|}(2^n)_{|\mathcal{V}_2|}} \left(1 - \frac{2q_d}{2^t}\right), \\
&\geq \left(1 - \frac{2q}{2^n}\right)^4 \left(1 - \frac{2q_d}{2^t}\right), \\
&\geq 1 - \frac{8q}{2^n} - \frac{2q_d}{2^t}
\end{aligned} \tag{2}$$

since $|\mathcal{V}_1|, |\mathcal{V}_2| \leq 2q$. Plugging (1) and (2) to Lemma 1, we conclude that

$$\left\|\mathsf{p}_{\mathcal{S}_0}^{q+v}(\cdot) - \mathsf{p}_{\mathcal{S}_1}^{q+v}(\cdot)\right\| \leq \frac{8q}{2^n} + \frac{2q_d}{2^t}.$$

4.2 Proof of Theorem 2

As in the proof of Theorem 1, we replace E_K and $E_{K'}^{-1}$ into P_1 and P_2, respectively. Precisely, the cost of this replacement is upper bounded by

$$2\mathbf{Adv}_E^{\mathsf{prp}}(3q, \mathsf{tm} + 3qc)$$

since $\mathcal{A}$ makes at most $3q$ block cipher queries.

Let $\mathsf{Enc}[P_1, P_2]$ denote the construction instantiated with permutations P_1 and P_2, and $\mathsf{Dec}[P_1, P_2]$ denotes the corresponding decryption oracle. Consider any adversary $\mathcal{A}$ interacts with either $\mathcal{S}_{\text{real}} = (\mathsf{Enc}, \mathsf{Dec})$ or $\mathcal{S}_{\text{ideal}} = (\$, \bot)$ where $\$$ denotes a truly random oracle with domain $\mathcal{N} \times \mathcal{H} \times \mathcal{M}$ that returns a tuple of a $|M|$-bit random string C and a t-bit random string T, and $\bot$ is an oracle that always returns 0. We assume without loss of generality that $\mathcal{A}$ makes encryption queries first and then makes decryption queries.

For the sake of our proof, we define a variant of the real world that outputs a tag without truncation for encryption queries denoted as $\widehat{\mathsf{Enc}}$ while the decryption oracle is not changed. We define an intermediate world that idealizes $\widehat{\mathcal{S}}_{\text{real}} = (\widehat{\mathsf{Enc}}, \mathsf{Dec})$ as a pair of (random) oracles $\widehat{\mathcal{S}}_{\text{inter}} = (\widehat{\$^*}, \bot)$ where $\widehat{\$^*}$ stands for encryption queries of the form (N, A, M) and $\bot$ is equivalent to that of the ideal world. $\widehat{\$^*}$ takes (N, A, M) and output (C, T') where C is a uniformly randomly chosen string of length $|M|$ (with replacement) and T' is chosen uniformly randomly from $\{0,1\}^n$ *without replacement if M is the same*. Informally, the oracle $\widehat{\$^*}$ reflects the property that $\widehat{\mathsf{Enc}}$ outputs different tags for queries with the same message. Moreover, let $\$^*$ be a variant of $\widehat{\$^*}$ that truncates the first $(n-t)$-bit of T' and the corresponding oracle $\mathcal{S}_{\text{inter}} = (\$^*, \bot)$. For any systems $\mathcal{S}_0$ and $\mathcal{S}_1$, let $\|\mathcal{S}_0 - \mathcal{S}_1\|$ denote the statistical distance between $\mathcal{S}_0$ and $\mathcal{S}_1$. Then, one has the following lemmas:

Lemma 2.

$$\|\mathcal{S}_{real} - \mathcal{S}_{inter}\| \leq \left\|\widehat{\mathcal{S}}_{real} - \widehat{\mathcal{S}}_{inter}\right\|.$$

Proof. Adversary $\mathcal{A}$ runs $\mathcal{A}$ and provides the access to its oracles. For the first oracle (Enc or $\$^*$), $\mathcal{A}$ gives $\mathcal{A}$ a truncation of tag which $\mathcal{A}$ is received. Finally, when $\mathcal{A}$ outputs its guess, adversary $\mathcal{A}$ will output the same guess. □

Lemma 3.

$$\|\mathcal{S}_{inter} - \mathcal{S}_{ideal}\| \leq \frac{q_e}{2^{n-\frac{t}{2}}}.$$

Proof. This result follows directly from the PRF security (indistinguishability from a uniformly random distribution) of Truncated Random Permutation (TRP) [GGM18] which states that the statistical distance between the p outputs of TRP and p outputs from a uniformly random distribution for t ($\leq n$)-bit values is upper bounded by $\frac{p}{2^{n-\frac{t}{2}}}$. The only difference lies in the output distributions of T' between $\$^*$ and $\$$, and the distribution of T' from $\$^*$ is (almost) identical to that of TRP. More precisely, given a transcript $(N_i, A_i, M_i, C_i, T_i')_{i\in[1..q]}$ from $\$^*$ and an index set I such that $i, j \in I \Leftrightarrow M_i = M_j$, the probability of obtaining $(T_i')_{i\in I}$ from $\$^*$ is exactly the same as the probability of obtaining $(T_i')_{i\in I}$ by querying TRP. It is independent of choices of T_j' for $j \notin I$. Hence, the probability of obtaining $(T_i')_{i\in[1..q]}$ from $\$^*$ is exactly the same as the probability of obtaining $(T_i')_{i\in[1..q]}$ by querying multiple independent of TRPs by partitioning $[1..q]$ into disjoint index sets I_k for some k such that $i, j \in I_k \Leftrightarrow M_i = M_j$. □

By triangle inequality, Lemma 2 and Lemma 3, we have

$$\begin{aligned}\|\mathcal{S}_{\mathrm{real}} - \mathcal{S}_{\mathrm{ideal}}\| &\le \|\mathcal{S}_{\mathrm{real}} - \mathcal{S}_{\mathrm{inter}}\| + \|\mathcal{S}_{\mathrm{inter}} - \mathcal{S}_{\mathrm{ideal}}\|, \\ &\le \left\|\widehat{\mathcal{S}}_{\mathrm{real}} - \widehat{\mathcal{S}}_{\mathrm{inter}}\right\| + \frac{q_e}{2^{n-\frac{t}{2}}}. \end{aligned} \tag{3}$$

Now, we prove the following lemma.

Lemma 4.

$$\left\|\widehat{\mathcal{S}}_{real} - \widehat{\mathcal{S}}_{inter}\right\| \le \frac{12q_e}{2^n} + \frac{7q_d}{2^t}.$$

The proof of Lemma 4 is deferred to the last of the section. By (3) and Lemma 4, it concludes the proof by

$$\|\mathcal{S}_{\mathrm{real}} - \mathcal{S}_{\mathrm{ideal}}\| \le \frac{(12 + 2^{\frac{t}{2}})q_e}{2^n} + \frac{7q_d}{2^t}.$$

Proof (of Lemma 4). Let Θ be the set of all attainable transcripts in the ideal world and $\tau = (\tau_e, \tau_d) \in \Theta$ be a transcript where τ_e and τ_d denote the list of encryption queries and the list of decryption queries, i.e.,

$$\begin{aligned}\tau_e &= \{(N_1, A_1, M_1, C_1, T_1), \ldots, (N_{q_e}, A_{q_e}, M_{q_e}, C_{q_e}, T_{q_e})\}, \\ \tau_d &= \{(N_{q_e+1}, A_{q_e+1}, C_{q_e+1}, T_{q_e+1}, b_{q_e+1}), \ldots, (N_{q_e+q_d}, A_{q_e+q_d}, C_{q_e+q_d}, T_{q_e+q_d}, b_{q_e+q_d})\}.\end{aligned}$$

Note that $|T_i| = n$ for $i \in [1..q_e]$, and $|T_i| = t$ otherwise. We assume that the adversary does not make redundant queries. More precisely, after receiving (N, A, M, C, T) from an encryption query, it does not query $(N, A, C, \mathsf{msb}_t(T))$ to the decryption oracle.

Additional Information and Extended Transcripts. At the end of the query phase, the oracles give additional information. For simplicity, we denote

$$N_{0,i} = \mathtt{pad}_n(N_i \,\|\, A_i \,\|\, b0), \;\; N_{1,i} = \mathtt{pad}_n(N_i \,\|\, A_i \,\|\, b1).$$

In the world $\widehat{\mathcal{S}}_{\mathrm{real}}$, the oracle additionally gives

$$S[N_i, A_i] \coloneqq P_1(N_{0,i}) \oplus P_1(N_{1,i})$$

for $i \in [1..q_e + q_d]$. In the world $\widehat{\mathcal{S}}_{\mathrm{inter}}$, for $i \in [1..q_e]$, it computes $\mathsf{msb}_{|M_i|}(S[N_i, A_i]) = C_i \oplus M_i$ and $\mathsf{lsb}_{n-|M_i|}(S[N_i, A_i])$ is uniformly randomly sampled. For each $i \in [q_e + 1..q_e + q_d]$, if $S[N_i, A_i]$ is undefined, assign a uniformly random value from $\{0,1\}^n$. In this way, for $i \in [q_e + 1..q_e + q_d]$, one can compute $M_i = \mathsf{msb}_{|C_i|}(S[N_i, A_i]) \oplus C_i$ and define τ_a as the set of M_i for $i \in [q_e + 1..q_e + q_d]$.

We will identify $\mathcal{V}_1 = \{P_1(N_{0,i})\} \cup \{P_1(N_{1,i})\}$ as a set of variables. $\mathcal{V}_2$ is identified as a set of variables such that

$$\mathcal{V}_2 = \{P_2(T_i)\}_{i \in [1..q_e]} \cup \left\{P_2(\langle 0 \rangle_{n-t} \,\|\, T_i), \ldots, P_2(\langle 2^{n-t} - 1 \rangle_{n-t} \,\|\, T_i)\right\}_{i \in [q_e+1..q_e+q_d]},$$

where $\langle j \rangle_{n-t}$ denotes $n - t$ bits representation of j for $0 \le j \le 2^{n-t} - 1$. We also define $\mathcal{V} = \mathcal{V}_1 \sqcup \mathcal{V}_2$. For simplicity, we denote $M'_i = \mathtt{pad}(M_i)$ for $i \in [1..q_e + q_d]$. Then we can construct the system of equations and inequalities Γ_τ as follows: For $i \in [1..q_e + q_d]$, $P_1(N_{0,i}) \oplus P_1(N_{1,i}) = S[N_i, A_i]$, and $P_1(N_{0,i}) \oplus P_1(T_i) = M'_i$ for $i \in [1..q_e]$ and $P_1(N_{0,i}) \oplus P_2(\langle j \rangle_{n-t} \,\|\, T_i) \ne M'_i$ where $j = 0, \ldots, 2^{n-t} - 1$ for $i \in [q_e + 1..q_d]$. To satisfy the conditions in Theorem 5, we must first define bad events on a transcript τ, and then we can apply the extended Mirror theory to each transcript that the bad event does not happen.

Defining and Bounding Bad Events. A transcript $\tau = (\tau_e, \tau_d, \tau_a)$ is defined as *bad* if one of the following conditions holds.

- $\mathsf{bad}_1 \Leftrightarrow$ there exists $(i_1, \ldots, i_n) \in [1..q_e]^{*n}$ such that $T_{i_1} = \cdots = T_{i_n}$.
- $\mathsf{bad}_2 \Leftrightarrow$ there exists $i \in [1..q]$ such that $S[N_i, A_i] = 0^n$.
- $\mathsf{bad}_3 \Leftrightarrow$ there exists $(i, j) \in [1..q_e]^{*2}$ such that
$$(T_i = T_j) \wedge (M'_i \oplus M'_j \in \{0^n, S[N_i, A_i], S[N_j, A_i], S[N_i, A_i] \oplus S[N_j, A_i]\}).$$

If a transcript τ is not bad, then it will be called a *good* transcript. The probability that the bad event occurs is obtained as follows:

- Since $q_e \leq 2^{n-1}$, one has
$$\Pr[\mathsf{bad}_1] \leq \frac{\binom{q_e}{n}}{(2^n)^{n-1}} \leq \left(\frac{q_e}{2^n}\right)^{n-1} \leq \frac{q_e}{2^n},$$
$$\Pr[\mathsf{bad}_2] \leq \frac{q_e + q_d}{2^n}.$$
- Without loss of generality, we fix $i < j$. For a such (i, j), we see that
$$\Pr[T_i = T_j] \leq \frac{1}{2^n}.$$
Furthermore, by the construction of $\widehat{\mathcal{S}}_{\text{inter}}$, $T_i = T_j$ implies $M'_i \neq M'_j$.
  - Case $M'_i \oplus M'_j = 0$: one has
$$\Pr\left[(M'_i \oplus M'_j = 0) \wedge (T_i = T_j)\right] = 0$$
since $\Pr\left[M'_i \oplus M'_j = 0 \mid T_i = T_j\right] = 0$.
  - Case $M'_i \oplus M'_j = S[N_i, A_i]$: For each i, there is at most one j such that $M'_i \oplus M'_j = S[N_i, A_i]$ and $T_i = T_j$ because if there exists k such that $T_i = T_j = T_k$, then $M'_k \neq M'_j \Rightarrow M'_i \oplus M'_k \neq M'_i \oplus M'_j = S[N_i, A_i]$. It concludes that
$$\Pr\left[(M'_i \oplus M'_j = S[N_i, A_i]) \wedge (T_i = T_j)\right] \leq \frac{1}{2^n}$$
for at most q_e number of (i, j) pairs since $\Pr\left[M'_i \oplus M'_j = S[N_i, A_i] \mid T_i = T_j\right] = 1$. Otherwise, the probability becomes 0.
  - Case $M'_i \oplus M'_j = S[N_j, A_j]$: Since $S[N_j, A_j]$ is chosen uniformly at random, one has
$$\Pr\left[M'_i \oplus M'_j = S[N_j, A_j]\right] = \frac{1}{2^n}.$$
Hence, we have
$$\Pr\left[M'_i \oplus M'_j = S[N_j, A_j] \mid T_i = T_j\right] \leq \frac{1}{2^{2n}}.$$
  - Case $M'_i \oplus M'_j = S[N_i, A_i] \oplus S[N_j, A_j]$: This has the same probability as the previous case of $M'_i \oplus M'_j = S[N_j]$.

  To conclude, one has
$$\Pr[\mathsf{bad}_3] \leq 0 + \frac{q_e}{2^n} + \frac{2q_e^2}{2^{2n}} \leq \frac{2q_e}{2^n},$$
  where the last inequality comes from $q_e \leq 2^{n-1}$.

By summing all the cases, we have
$$\Pr[\mathsf{bad}] \leq \Pr[\mathsf{bad}_1] + \Pr[\mathsf{bad}_2] + \Pr[\mathsf{bad}_3] \leq \frac{4q_e + q_d}{2^n}. \tag{4}$$

Good Transcript Analysis. For a good transcript τ and its system Γ_τ, by assuming nonces are not repeated, we observe that

- Γ_τ is nice by $\neg(\mathsf{bad}_2 \wedge \mathsf{bad}_3)$;
- $\xi_{\max} \leq 2n-1$ and $\xi_{\max}^2 n + \xi_{max} \leq 4n^3 \leq 2^{n/2}$ since $n \geq 36$ by $\neg\mathsf{bad}_1$.

Henceforth, we can apply recent result of Mirror theory [CLL24] and then we have

$$h(\Gamma_\tau) \geq \frac{(2^n-2)_{|\mathcal{V}_1|}(2^n-2)_{|\mathcal{V}_2|}}{2^{n(2q_e+a)}}\left(1-\frac{2q_d}{2^t}\right),$$

where a is the number of simulated $S[N_i, A_i]$ for $i \in [q_e+1..q_e+q_d]$. Note that $2q_e+a$ is the exact number of equations from a given extended transcript τ. For the completeness of the proof, we restate the Mirror theory in Theorem 5. Furthermore, we see that

$$\mathsf{p}_{\mathcal{S}_0}^{q_e+q_d}(\tau) = \frac{1}{2^{n(2q_e+a)}},\ \mathsf{p}_{\mathcal{S}_1}^{q_e+q_d}(\tau) = \frac{h(\Gamma_\tau)}{(2^n)_{|\mathcal{V}_1|}(2^n)_{|\mathcal{V}_2|}}.$$

From the above, one has

$$\frac{\mathsf{p}_{\mathcal{S}_1}^{q_e+q_d}(\tau)}{\mathsf{p}_{\mathcal{S}_0}^{q_e+q_d}(\tau)} \geq \frac{(2^n-2)_{|\mathcal{V}_1|}(2^n-2)_{|\mathcal{V}_2|}}{(2^n)_{|\mathcal{V}_1|}(2^n)_{|\mathcal{V}_2|}}\left(1-\frac{2q_d}{2^t}\right) \geq 1-\frac{8q_e}{2^n}-\frac{6q_d}{2^t} \tag{5}$$

since $t \leq n$ and $|\mathcal{V}_1|, |\mathcal{V}_2| \leq 2q_e + a \leq 2q_e + 2^{n-t} \cdot q_d$. Plugging (4) and (5) to Lemma 1, we conclude that

$$\left\|\widehat{\mathcal{S}}_{\text{real}} - \widehat{\mathcal{S}}_{\text{inter}}\right\| \leq \frac{12q_e}{2^n} + \frac{7q_d}{2^t}. \qquad \square$$

5 Implementations

5.1 Benchmark Settings

Benchmarking methodologies. To assess the efficiency of Cymric compared to other AE modes, we conducted a benchmark where all modes were instantiated with AES-128 as the underlying block cipher. This approach ensures that the comparison focuses solely on the performance of the operating modes, as they share a common cryptographic primitive. Given that Cymric is designed for short inputs, it is particularly relevant to compare it against lightweight AE algorithms. To this end, we conducted a supplemental benchmark against several finalists from the NIST LwC project, including Ascon [DEMS19] which has been selected for standardization. While this second benchmark offers less direct insight into the operating modes themselves–owing to the heterogeneous nature of the underlying primitives, with some algorithms being permutation-based designs and others relying on block ciphers–it enables a meaningful comparison against state-of-the-art lightweight AE schemes.

Input lengths. Given Cymric1 and Cymric2 have different limitations on the maximum input lengths they can handle, two distinct scenarios for different payloads were assessed for benchmarking purposes. Even though this does not affect Cymric in terms of calculations, which requires three block cipher calls unconditionally, some of the schemes selected for comparison show performance fluctuations based on the input size. Inputs in favor of competitors were prioritized, notably a payload with a 96-bit nonce to avoid two extra GHASH calls in GCM, and another payload without additional data to save one GHASH call in both GCM and AES-GCM-SIV as well as a block cipher and a permutation call in OCB and Ascon, respectively. Scenario 1 handles $(\nu, \alpha, \ell) = (96, 24, 32)$ as these parameters

represent the longest multiple of bytes that Cymric1 can handle with 96-bit nonces, while Scenario 2 handles $(\nu, \alpha, \ell) = (120, 0, 120)$ to cover larger inputs with Cymric2. Note that for modes that do not handle custom nonce length (e.g., AES-GCM-SIV supports 96-bit nonces only), ν is not affected.

Platforms. Since Cymric is tailored for short payloads, which are particularly relevant in embedded systems, we present benchmarks on 8-bit AVR ATmega128 and 32-bit ARM Cortex-M4 microcontrollers to encompass platforms with varying computational capabilities. The ATmega128 microcontroller is built on an 8-bit AVR processor and includes 128 KB program flash, 4 KB SRAM and 4 KB EEPROM. It comes with 32 general-purpose registers and most instructions execute in a single cycle. Noteworthy observations regarding cryptographic implementations are that the multiplication instruction takes two cycles to multiply 8-bit operands and that the bitwise shift instruction is restricted to shifting one bit at a time. The platform is simulated using Microchip Studio `7.0.2594` to take advantage of the cycle counter, which is only available with the simulator. The code was compiled with `avr-gcc 14.1.0` using the `-O3` optimization flag. Regarding 32-bit ARM, we use the STM32F407VG development board which features a Cortex-M4 running up to 168 MHz, 1024 KB of flash memory and 192 KB of RAM. The ARMv7-M architecture includes sixteen 32-bit registers and employs a 3-stage pipeline. Most instructions take a single cycle, except branches and memory accesses that may take more cycles. Thanks to the barrel shifter, flexible second operands can be shifted or rotated as part of the instruction without any impact on performance. For our benchmark, the clock is set to 24 MHz to execute code from flash with zero-wait state. All implementations are compiled using `arm-none-eabi-gcc 14.2.1` along with the `-O3` optimization flag. Clock cycles are measured using the `DWT_CYCCNT` cycle counter register while stack usage and code size are measured using scripts which leverage GNU binutils tools with call-graph tracing.

5.2 Instantiation with AES-128

Reference modes. The modes selected for comparison are GCM [Mor07] and OCB [KR11], as both are well-studied and have variants that provide higher security, namely AES-GCM-SIV [GLL19] and XOCB [BHI+23], enabling a fair comparison in terms of both performance and security. Although GCM is not commonly used in IoT protocols due to the need of implementing both the block cipher and the Galois field multiplication, it remains a valuable point of reference for benchmarking as it requires few block cipher calls when processing short inputs. While CCM [Dwo07] is commonly favored in IoT protocols (e.g. Bluetooth [PBB+17], LoRa [AF18], Sigfox [Fer20]), OCB offers better performance (i.e. fewer block cipher calls) than CCM as well as support for parallelization, making it a more relevant candidate for our benchmarks. Only encryption routines are benchmarked as all modes are inverse-free — except XOCB, which requires the block cipher inverse operation only when processing full message blocks. To establish fairness in comparison with Cymric, all modes were implemented so that the length of messages and additional data is limited to at most one block. This approach can lead to a smaller footprint by eventually simplifying the code and removing calculations that are not necessary when inputs are smaller than the block size. Optimizations aimed at improving performance are considered whenever possible, provided they have a reasonable effect on memory usage. For instance, both GCM and OCB generate subkey material by encrypting the null block, and the implementations hereafter considered for benchmark pre-compute these values to save one block cipher call, extending the key size by one block. For more details on our implementation choices, we refer the interested reader to our publicly available online code at https://github.com/aadomn/cymric.

Table 2: Benchmark of various AE modes all instantiated with AES-128 as the underlying block cipher. The security column refers to bit security of NAE advantage (Def. 2) assuming PRP/SPRP for block cipher. The bit security of AES-GCM-SIV comes from [IS17, Th.3] with $R = 1$ and AES-128. Scenarios 1 and 2 handle $(\nu, \alpha, \ell) = (96, 24, 32)$ and $(120, 0, 120)$, respectively. Cymric1 is not considered for scenario 2 as it exceeds the maximum input length supported by the scheme. Bold values represent the best results for each category.

Platform	Mode	Security (bits)	Speed (cycles) Scenario 1	Speed (cycles) Scenario 2	Memory (bytes) Key	Memory (bytes) Stack	Memory (bytes) Code
ATmega128 (AVR)	Cymric1	**128**	9 881	-	32	**238**	**2 152**
	Cymric2	85.3	**9 687**	**10 084**	32	**238**	2 766
	XOCB	85.3	26 699	26 989	**16**	295	7 632
	AES-GCM-SIV	64	52 211	42 126	**16**	537	5 656
	OCB	64	12 871	10 910	32	270	7 378
	GCM	64	39 239	59 628	32	490	4 466
STM32F407 (Cortex-M4)	Cymric1	**128**	9 644	-	32	472	**3 246**
	Cymric2	85.3	9 584	9 697	32	**464**	3 648
	XOCB	85.3	17 676	17 942	**16**	648	5 348
	AES-GCM-SIV	64	20 775	19 472	**16**	788	5 102
	OCB	64	**8 533**	**8 599**	32	640	4 996
	GCM	64	9 917	11 306	32	676	4 314

Primitives' implementations. On AVR, we use the RijndaelFast implementation from [Poe07] for AES, which relies on so-called T-tables for efficient computation. Multiplications in $\mathsf{GF}(2^{128})$ are performed using Shoup's 4-bit tables method [Sho96], requiring 64 bytes of storage along with 256 bytes per key. Doublings, on the other hand, are handled through bit shifts and a conditional XOR operation based on the value of the most significant bit. On ARM, although the Cortex-M4 has no internal cache memory, a system-level cache can be incorporated into the system-on-chip design. When combined with tabled-based implementations, it would introduce timing leakages that could be leveraged to mount side-channel attacks. To mitigate this risk and broaden the scope of our benchmark, we adopt the fixsliced implementation strategy [AP21], which is currently the fastest constant-time AES implementation available for this platform. $\mathsf{GF}(2^{128})$ multiplications are implemented using the BearSSL carryless multiplications technique[5], whereas doublings simply involve bit shifts and a conditional XOR based on the most significant bit value. On both platforms, conditional XORs for doublings (which are only used by OCB and XOCB) are implemented in a constant-time fashion.

Results interpretation. As shown in Table 2, thanks to its simplicity Cymric achieves the best results in terms of footprint on both platforms, offering the smallest code size and stack consumption. In terms of performance, Cymric outperforms all other schemes on AVR and demonstrates performance comparable to GCM and OCB on ARM. This can be attributed to the fact that bitsliced AES on 32-bit platforms processes two blocks at a time due to its 8-bit S-box. Hence, even though OCB requires one more block cipher call than Cymric in scenario 1, this additional call can be parallelized at no extra cost with this implementation. However from a security-equivalent standpoint, Cymric2 remains about twice faster than XOCB. Bear in mind that, because the round keys are calculated on-the-fly, this negatively impacts the performance of Cymric, as other modes perform only a single key expansion—except for AES-GCM-SIV, which undergoes nonce-based rekeying. Therefore, hardcoding the precomputed round keys would give a slight additional advantage to Cymric, as the reported results account for both key expansions.

[5] https://www.bearssl.org/constanttime.html#ghash-for-gcm

5.3 Lightweight Cryptography

Reference AE schemes. We consider the following shortlist of NIST LwC finalists for comparison: Ascon, Xoodyak [DHP+19] and PHOTON-Beetle-AEAD [BCD+19] for their sponge-based designs, as well as GIFT-COFB [BCI+19] and Romulus-N [GIK+19] to represent (tweakable) block cipher modes. This selection offers a diverse range of AE schemes, with multiple security guarantees and implementation characteristics. For Ascon, we use Ascon-AEAD128 specified by the NIST draft standard [TMC+24], which is based on the variant Ascon-128a being the secondary recommendation from the original submission. This variant features a 128-bit rate and iterates its permutation for 8 rounds when processing associated data and plaintext. For PHOTON-Beetle-AEAD, we use the primary recommendation PHOTON-Beetle-AEAD[128] from the original submission which also features a 128-bit rate. Unlike the AES-based benchmark where different modes utilized the same block cipher, this comparison is more challenging as the results are heavily influenced by the different underlying primitives and the extent of optimization in their implementations. To attempt for a fair evaluation, we use implementations from reference packages when available (e.g., XKCP[6] for Xoodyak) or we leverage prior optimization efforts, particularly those from Weatherley[7], which are tailored for both ARMv7-M and AVR platforms.

Lightweight Cymric instantiations. To build a highly efficient instantiation of Cymric, one might be tempted to leverage its BBB security by combining it with a small (*i.e.*, 64 or 96-bit) block cipher to achieve competitive performance while offering decent security. However, such an approach would further restrict the limited number of bytes that could be processed, which is why we focus on 128-bit block ciphers instead. Besides efficiency, we prioritize ciphers that are well-established and offer a high security margin. With these factors in mind, we choose to instantiate Cymric with GIFT-128 [BPP+17] as it has undergone extensive cryptanalysis (*e.g.*, [SWW21, ZDC+21, SWW22, WLHL24]) and is also used by GIFT-COFB, a scheme included in our benchmark. Additionally, we benchmark another instantiation using LEA-128 [HLK+14], a 128-bit Addition-Rotation-XOR (ARX) cipher standardized by Korea (KS X 3246) [KSX16] and ISO/IEC 29192-2 [ISO19]. In addition to offering outstanding performance, especially on 32-bit processors, LEA-128 has undergone thorough cryptanalysis [SHY16, SWW17, DS18, ZGH16, KKS20, BA20, CBY23] and the best attacks [CBY23] reach only 17 out of 24 rounds. For GIFT, we use the bitsliced implementations introduced in [ANP20] while for LEA, we develop custom assembly implementations for both platforms. To ensure a fair comparison and achieve the best possible results for Cymric across the different metrics, we report results for both on-the-fly and precomputed key schedules. Note that on ARMv7-M, no results are reported for LEA-128 with precomputed round keys as our implementation unconditionally expands the key on-the-fly given the minimal overhead on this platform.

Results interpretation. As shown in Tables 3 and 4, LEA128-Cymric provides the best performance on both platforms and in both scenarios. The performance advantage over Ascon is primarily reflected in Scenario 1, with improvements of approximately 39% on ARM Cortex-M4 and 114% on AVR when using pre-computed round keys. Furthermore, Cymric delivers the smallest code size, a result of the simplicity of its mode, as also observed in the previous benchmark. Regarding GIFT128-Cymric, it outperforms Ascon by around 26% on AVR in Scenario 1 when using pre-computed round keys, but is slower otherwise. This disparity arises because the Ascon permutation is less suited to 8-bit platforms compared to GIFT, but better suited to 32-bit ones. However, in terms of

[6] https://github.com/XKCP/XKCP
[7] https://github.com/rweather/lwc-finalists

Table 3: Benchmark of lightweight AE schemes on AVR ATmega128. Scenarios 1 and 2 handle $(\nu, \alpha, \ell) = (96, 24, 32)$ and $(120, 0, 120)$, respectively. Cymric1 is not considered in Scenario 2 as it exceeds the maximum input length supported by the scheme. Bold values represent the best results for each category.

AEAD	Security (bits)	Implementation	Speed (cycles)		Memory (bytes)		
			Scenario 1	Scenario 2	Key	Stack	Code
LEA128-Cymric1	**128**	Ours	19 163	-	32	491	1 590
			11 276*	-	768*	107*	**1 128***
LEA128-Cymric2	85.3		19 305	19 400	32	488	2 208
			11 416*	**11 517***	768*	104*	1 746*
GIFT128-Cymric1	**128**		31 609	-	32	427	5 162
			19 139*	-	640*	107*	2 252*
GIFT128-Cymric2	85.3		31 764	31 842	32	423	5 780
			19 293*	19 379*	640*	104*	2 870*
Ascon-AEAD128	**128**	ascon/ascon-c	24 143	18 661	**16**	122	4 036
Xoodyak	**128**	rweather/lwc-finalists	43 441	43 640	**16**	**98**	2 542
Romulus-N	**128**	rweather/lwc-finalists	30 364	30 525	**16**	165	5 592
PHOTON-Beetle-AEAD[128]	121	rweather/lwc-finalists	60 357	40 675	**16**	131	7 840
GIFT-COFB	64	aadomn/gift	27 224	26 993	**16**	398	9 192

* Using pre-computed round keys.

Table 4: Benchmark of lightweight AE schemes on ARM Cortex-M4. Scenarios 1 and 2 handle $(\nu, \alpha, \ell) = (96, 24, 32)$ and $(120, 0, 120)$, respectively. Cymric1 is not considered in Scenario 2 as it exceeds the maximum input length supported by the scheme. Bold values represent the best results for each category.

AEAD	Security (bits)	Implementation	Speed (cycles)		Memory (bytes)		
			Scenario 1	Scenario 2	Key	Stack	Code
LEA128-Cymric1	**128**	Ours	2 274	-	32	160	**1 052**
LEA128-Cymric2	85.3		**2 198**	**2 316**	32	**152**	1 390
GIFT128-Cymric1	**128**		8 218	-	32	800	2 224
			4 500*	-	640*	160*	1 268*
GIFT128-Cymric2	85.3		8 157	8 276	32	792	2 582
			4 438*	4 560*	640*	**152***	1 606*
Ascon-AEAD128	**128**	ascon/ascon-c	3 054	2 457	**16**	160	1 368
Xoodyak	**128**	XKCP/XKCP	3 572	3 669	**16**	240	3 304
Romulus-N	**128**	aadomn/skinny	11 061	11 199	**16**	980	9 868
PHOTON-Beetle-AEAD[128]	121	rweather/lwc-finalists	30 897	20 702	**16**	284	6 746
GIFT-COFB	64	aadomn/gift	6 600	6 405	**16**	496	3 970

* Using pre-computed round keys.

memory consumption, it is worth noting that when Cymric is used with precomputed round keys, storage requirements can become significant, depending on the underlying block cipher. In the case of GIFT-128 and LEA-128, the storage of key material exceeds half a kilobyte due to their high number of rounds and the fact that Cymric requires two distinct keys. The impact on stack usage is less pronounced as the round keys used for the first two cipher calls can be overwritten by those used in the final call.

6 Impossibility of BBB-secure Short-input AE with Two Block Cipher Calls

We show that any (short-input) block cipher mode for efficient nonce-based AE within two n-bit block cipher calls provides $n/2$-bit security at best. We assume the standard model security and support of n-bit plaintext. This implies the optimality of Cymric2 in terms of the number of block cipher calls as a short-input BBB-secure AE.

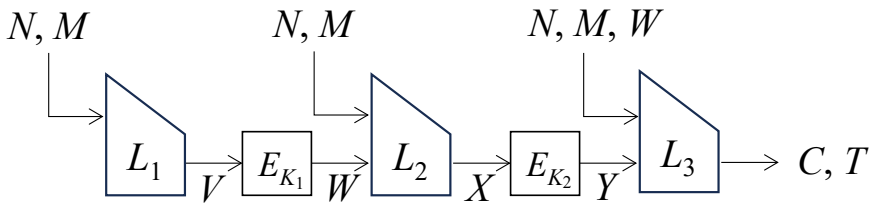

Figure 3: GAE with two block ciphers E_{K_1}, E_{K_2} and three linear functions L_1, L_2, L_3.

6.1 Generic Construction

AEs are mainly constructed from non-linear primitives, including block ciphers, full field multiplications, and linear operations such as XOR and field multiplications by constants, *e.g.*, 2 and 3 (which represents polynomial `x` and `x+1`). A full field multiplication often requires an equivalent implementation cost as a block cipher and are generally not suitable for IoT applications. Indeed, Cymric and all NIST LwC finalists do not use any costly operations except their cryptographic primitives, and several existing works for efficient symmetric-key design such as [NSS20, NSS21, SS23] considered generic constructions with this criterion to show the optimality of their proposals. With a similar motivation in mind, we define GAE, a generic construction of (short-input) AEs that mainly consists of linear operations and two block-cipher calls and that accepts ν-bit nonce and an n-bit plaintext, and returns an n-bit ciphertext. Let $E_{K_1} : \{0,1\}^n \to \{0,1\}^n$ and $E_{K_2} : \{0,1\}^n \to \{0,1\}^n$ be the underlying block ciphers with keys K_1 and K_2, respectively.

GAE uses three functions L_1, L_2, and L_3. The first function $L_1 : \{0,1\}^\nu \times \{0,1\}^n \to \{0,1\}^n$ is called before the first block cipher E_{K_1} and takes nonce N and a plaintext M, and returns a plaintext block V of E_{K_1}. The second function $L_2 : \{0,1\}^\nu \times \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ is called between E_{K_1} and E_{K_2}, and takes N, M and the ciphertext block $W = E_{K_1}(V)$, and returns a plaintext block $X \in \{0,1\}^n$ of E_{K_2}. The third function $L_3 : \{0,1\}^\nu \times \{0,1\}^n \times \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n \times \{0,1\}^t$ takes N, M, W, and the ciphertext block $Y = E_{K_2}(X)$, and returns a ciphertext C and a tag T of GAE. Using these functions, an encryption of GAE is defined as $\mathsf{GAE}.\mathcal{E}[E_{K_1}, E_{K_2}](N, M) = (C, T)$ such that

$$
\begin{aligned}
&V = L_1(N, M);\ W = E_{K_1}(V); && \text{(First block cipher call)}\\
&X = L_2(N, M, W);\ Y = E_{K_2}(X); && \text{(Second block cipher call)}\\
&(C, T) = L_3(N, M, W, Y). && \text{(Finalization)}
\end{aligned}
$$

Fig. 3 shows the structure of $\mathsf{GAE}.\mathcal{E}$. Let $\mathsf{GAE}.\mathcal{D}[E_{K_1}, E_{K_2}]$ be the decryption function such that for any $(N, M) \in \{0,1\}^\nu \times \{0,1\}^n$, if $(C, T) = \mathsf{GAE}.\mathcal{E}[E_{K_1}, E_{K_2}](N, M)$, then $\mathsf{GAE}.\mathcal{D}[E_{K_1}, E_{K_2}](N, C, T) = M$.

Next, we show specifications of the functions L_1, L_2, and L_3. Let $v_1, v_2 \in \{0,1\}^n$ be coefficients of L_1 and $\mathrm{Const}_1 \in \{0,1\}^n$ a constant value. Let x_1, x_2, x_3 be coefficients of L_2 and Const_2 a constant value. Let $c_1, c_2, c_3, c_4, t_1, t_2, t_3, t_4$ be coefficients of L_3 and $\mathrm{Const}_3 \in \{0,1\}^n, \mathrm{Const}_4 \in \{0,1\}^t$ constant values. For each coefficient $a \in \{v_1, x_1, c_1, t_1\}$ and an input $Z \in \{0,1\}^\nu$, $a \cdot Z$ represents a function from $\{0,1\}^\nu$ to $\{0,1\}^n$. Similarly, for each $a \in \{v_2, x_2, x_3, c_2, c_3, c_4, t_2, t_3, t_4\}$ and input $Z \in \{0,1\}^n$, $a \cdot Z$ represents a function from $\{0,1\}^n$ to $\{0,1\}^n$. For a coefficient a, if $a = 0$, then for any input Z, $a \cdot Z := 0^n$. We assume that for each $a \in \{c_3, c_4, t_3, t_4\}$, the function with a is linear, which define T or C, i.e., for each input $Z_1, Z_2 \in \{0,1\}^n$, $a \cdot Z_1 \oplus a \cdot Z_2 = a \cdot (Z_1 \oplus Z_2)$ holds.[8] For

[8] Note that we do not assume the linearity of the functions with $\{v_1, v_2, x_1, x_2, x_3, c_1, c_2, t_1, t_2\}$, since these functions may contain efficient but non-linear functions such as padding functions. For example, the one-zero padding function $\mathtt{pad}_n$, which is commonly used in symmetric-key algorithms, is not linear: for inputs Z and Z' with $|Z| = |Z'|$, $\mathtt{pad}_n(Z) \oplus \mathtt{pad}_n(Z') = (Z \,\|\, 10^{n-|Z|-1}) \oplus (Z' \,\|\, 10^{n-|Z|-1}) = (Z \oplus Z') \,\|\, 0^{n-|Z|} \neq \mathtt{pad}_n(Z \oplus Z')$.

each $a \in \{c_3, c_4, t_3, t_4\}$, a^{-1} represents the inverse function of a, i.e., for an input Z, $a^{-1} \cdot (a \cdot Z) = Z$ and $a \cdot (a^{-1} \cdot Z) = Z$. Regarding the other coefficients a, we assume that given an output Z', each of inputs Z such that $Z' = a \cdot Z$ is efficiently computable. For nonce $N \in \{0,1\}^\nu$ and a plaintext $M \in \{0,1\}^n$, the functions are defined as follows.

$$
\begin{aligned}
&L_1(N, M) = V = v_1 \cdot N \oplus v_2 \cdot M \oplus \mathrm{Const}_1.\\
&L_2(N, M, W) = X = x_1 \cdot N \oplus x_2 \cdot M \oplus x_3 \cdot W \oplus \mathrm{Const}_2.\\
&L_3(N, M, W, Y) = (C, T), \text{ where}\\
&\quad C = c_1 \cdot N \oplus c_2 \cdot M \oplus c_3 \cdot W \oplus c_4 \cdot Y \oplus \mathrm{Const}_3, \text{ and}\\
&\quad T = \mathsf{lsb}_t(t_1 \cdot N \oplus t_2 \cdot M \oplus t_3 \cdot W \oplus t_4 \cdot Y) \oplus \mathrm{Const}_4.
\end{aligned}
$$

Relation between Cymric and GAE. We discuss the relation between Cymric and GAE. Cymric consists of three block cipher invocations, XOR, and functions that define inputs to block ciphers and those define the tag T and the ciphertext C taking outputs of the block cipher. The functions inside Cymric which define T and C are linear. There is no restriction on the other functions that define inputs to block ciphers. Hence, all functions in Cymric with empty associated data A meet the conditions on the functions of GAE.

6.2 Birthday Attack on GAE with $t = n$

The following theorem shows that there exists an attack on GAE with $t = n$ with $O(2^{n/2})$ encryption queries.

Theorem 3. *Assume that $t = n$. Then, there exists a $(q_e, 0, O(q_e^2))$-adversary $\mathcal{A}$ on* GAE *such that* $\mathbf{Adv}^{\mathsf{nAE}}_{\mathsf{GAE}[E_{K_1}, E_{K_2}]}(\mathcal{A}) = O\left(\frac{q_e^2}{2^n}\right)$.

Proof. Without loss of generality, we assume that $c_1 = 0$, $c_2 = 0$, $t_1 = 0$, $t_2 = 0$, $\mathrm{Const}_3 = 0$, and $\mathrm{Const}_4 = 0$, since the constants, nonces, and plaintexts are public for the adversary and do not affect the security of GAE.

We consider the following cases. Let $\mathsf{Fin} := \begin{pmatrix} c_3 & c_4 \\ t_3 & t_4 \end{pmatrix}$.

$\mathsf{rank}(\mathsf{Fin}) \leq 1$. In the real world, there exists $r \in \{0,1\}^n$ such that for any (N, M), $r \cdot C = T$. On the other hand, in the ideal world, for any $r \in \{0,1\}^n$, we have $\Pr[r \cdot C = T] = \frac{1}{2^n}$, since C and T are independently chosen. With the difference, one can construct a $(1, 0, O(1))$-adversary $\mathcal{A}$ such that $\mathbf{Adv}^{\mathsf{nAE}}_{\mathsf{GAE}[E_{K_1}, E_{K_2}]}(\mathcal{A}) = O(1)$.

$\mathsf{rank}(\mathsf{Fin}) = 2 \wedge \Big(\exists (\boldsymbol{N_1}, \boldsymbol{M_1}), (\boldsymbol{N_2}, \boldsymbol{M_2})$ s.t. $(\boldsymbol{N_1}, \boldsymbol{M_1}) \neq (\boldsymbol{N_2}, \boldsymbol{M_2}) \wedge \boldsymbol{V_1} = \boldsymbol{V_2}\Big)$. With the distinct pairs (N_1, M_1) and (N_2, M_2) such that $V_1 = V_2$, one has the collision $W_1 = W_2$ in the real world, which can be confirmed by recovering W_1 and W_2 from the equations $C_i = c_3 \cdot W_i \oplus c_4 \cdot Y_i$ and $T_i = t_3 \cdot W_i \oplus t_4 \cdot Y_i$, where $i \in \{1, 2\}$. In the ideal world, we have $\Pr[W_1 = W_2] \leq \frac{1}{2^n}$. With the equation, we can construct a $(2, 0, O(1))$-adversary $\mathcal{A}$ such that $\mathbf{Adv}^{\mathsf{nAE}}_{\mathsf{GAE}[E_{K_1}, E_{K_2}]}(\mathcal{A}) = O(1)$.

$\mathsf{rank}(\mathsf{Fin}) = 2 \wedge \Big(\forall (\boldsymbol{N_1}, \boldsymbol{M_1}), (\boldsymbol{N_2}, \boldsymbol{M_2})$ s.t. $(\boldsymbol{N_1}, \boldsymbol{M_1}) \neq (\boldsymbol{N_2}, \boldsymbol{M_2}) : \boldsymbol{V_1} \neq \boldsymbol{V_2}\Big)$. In this case, we define an adversary $\mathcal{A}$ in the following.

1 Choose q_e pairs of nonce and plaintext $(N_1, M_1), \ldots, (N_{q_e}, M_{q_e})$ such that $V_1, \ldots, V_{q_e}$ are all distinct.

2 For $i \in [1..q_e]$, make an encryption query (N_i, M_i) and receive the pair (C_i, T_i).

3 For $i \in [1..q_e]$, recover W_i by solving the equations $C_i = c_3 \cdot W_i \oplus c_4 \cdot Y_i$; $T_i = t_3 \cdot W_i \oplus t_4 \cdot Y_i$.

4 If $\exists i, j \in [1..q_e]$ s.t. $i \neq j \wedge W_i = W_j$, then return 0; Otherwise return 1.

In the real world, $V_1, \ldots, V_{q_e}$ are all distinct, ensuring that $W_1, \ldots, W_{q_e}$ are all distinct and the adversary returns 1. In the ideal world, since $C_1, T_1, \ldots, C_{q_e}, T_{q_e}$ are independently chosen, the probability that the adversary returns 0 is $O\left(\frac{q_e^2}{2^n}\right)$. Hence, there exists a $(q_e, 0, O(q_e^2))$-adversary $\mathcal{A}$ on GAE such that $\mathbf{Adv}^{\mathsf{nAE}}_{\mathsf{GAE}[E_{K_1}, E_{K_2}]}(\mathcal{A}) = O\left(\frac{q_e^2}{2^n}\right)$.

□

6.3 Birthday Attack on GAE with $t < n$

The following theorem shows that there exists an attack on GAE with $t < n$ with $2^{n/2}$ encryption queries.

Theorem 4. *There exists a $(q_e, q_d, O(q_e^2))$-adversary $\mathcal{A}$ on GAE such that $q_d \leq 1$ and $\mathbf{Adv}^{\mathsf{nAE}}_{\mathsf{GAE}[E_{K_1}, E_{K_2}]}(\mathcal{A}) = O\left(\frac{q_e^2}{2^n}\right)$.*

Proof (overview). Without loss of generality, we assume that $c_1 = 0$, $c_2 = 0$, $t_1 = 0$, $t_2 = 0$, $\mathrm{Const}_3 = 0^n$, and $\mathrm{Const}_4 = 0^t$, since the constants, nonces, and plaintexts are public for the adversary and do not affect the security of GAE.

If $\mathrm{rank}(\mathsf{Fin}) \leq 1$, then as the proof of Theorem 3, one can construct a $(1, 0, O(1))$-adversary $\mathcal{A}$ such that $\mathbf{Adv}^{\mathsf{nAE}}_{\mathsf{GAE}[E_{K_1}, E_{K_2}]}(\mathcal{A}) = O(1)$. For $\mathrm{rank}(\mathsf{Fin}) = 2$, since $t < n$, we cannot recover ciphertext blocks W_i and Y_i from the pair of the ciphertext C_i and the tag T_i as the proof of Theorem 3. However, we can confirm collisions for W_i and for Y_i by using the following conditions. With the conditions, we can obtain the bound in Theorem 4. The full proof is given in Appendix B.

$$\begin{aligned}
&\text{If } c_4 \neq 0, \text{ then } W_i = W_j \Leftrightarrow (t_4 \cdot c_4^{-1} \cdot c_3) \cdot W_i = (t_4 \cdot c_4^{-1} \cdot c_3) \cdot W_j \\
&\qquad\qquad \Rightarrow \mathsf{lsb}_t(t_4 \cdot c_4^{-1} \cdot C_i) \oplus T_i = \mathsf{lsb}_t(t_4 \cdot c_4^{-1} \cdot C_j) \oplus T_j. \\
&\text{If } c_4 = 0 \wedge c_3 \neq 0, \text{ then } W_i = W_j \Leftrightarrow C_i = C_j. \\
&\text{If } c_3 \neq 0, \text{ then } Y_i = Y_j \Leftrightarrow (t_3 \cdot c_3^{-1} \cdot c_4) \cdot Y_i = (t_3 \cdot c_3^{-1} \cdot c_4) \cdot Y_j \\
&\qquad\qquad \Rightarrow \mathsf{lsb}_t(t_3 \cdot c_3^{-1} \cdot C_i) \oplus T_i = \mathsf{lsb}_t(t_3 \cdot c_3^{-1} \cdot C_j) \oplus T_j. \\
&\text{If } c_3 = 0 \wedge c_4 \neq 0, \text{ then } Y_i = Y_j \Leftrightarrow C_i = C_j.
\end{aligned}$$

□

7 Conclusion

We have presented a family of short-input AE block cipher modes, Cymric. This is the first beyond-birthday-bound (BBB) secure AE explicitly focusing on short inputs, a critical aspect in practical IoT-like applications where existing constructions fine-tuned for tiny inputs are limited to birthday-bound security. We provided a comprehensive benchmark on microcontrollers and theoretical evidence showing that the cost of Cymric (in particular Cymric2), which involves three block cipher calls, is indeed minimal to achieve BBB security. The structure of Cymric is an amalgamation of SoP PRF and EWCDM MAC, but dedicated proofs were necessary. We introduced a new technique for proving BBB security —utilizing unique intermediate worlds combined with Lemmas 2, 3, and 4— which may be of independent interest.

Several future directions could be considered. Optimizing Cymric, say by reducing the number of keys, is a promising direction. As mentioned in the Introduction, extending constructions based on cryptographic permutations or (short) tweakable block ciphers or forkciphers would be interesting. In a broader context, it is also worth further exploring the

gap between general-purpose AEs and short-input ones in terms of security, efficiency, and input ranges. Finally, as discussed in the introduction, evaluating the concrete benefits of Cymric (or more generally, BBB-secure short-input AEs) in real-world applications requires further study. We hope that our work will stimulate research in these directions.

Acknowledgment

We thank the anonymous reviewers and our shepherd for their insightful feedback and suggestions. Wonseok Choi was supported in part by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. RS-2023-00239620), by AnalytiXIN and by Sunday Group.

References

[AF18] Gildas Avoine and Loïc Ferreira. Rescuing LoRaWAN 1.0. In *Financial Cryptography and Data Security: FC 2018*, page 253–271, Berlin, Heidelberg, 2018. Springer-Verlag.

[ALP+19a] Elena Andreeva, Virginie Lallemand, Antoon Purnal, Reza Reyhanitabar, Arnab Roy, and Damian Vizár. ForkAE. A submission to NIST Lightweight Cryptography, 2019.

[ALP+19b] Elena Andreeva, Virginie Lallemand, Antoon Purnal, Reza Reyhanitabar, Arnab Roy, and Damian Vizár. Forkcipher: A new primitive for authenticated encryption of very short messages. In Steven D. Galbraith and Shiho Moriai, editors, *ASIACRYPT 2019, Part II*, volume 11922 of *LNCS*, pages 153–182. Springer, Cham, December 2019.

[AMS23] Alexandre Adomnicai, Kazuhiko Minematsu, and Junji Shikata. Authenticated encryption for very short inputs. In Mike Rosulek, editor, *CT-RSA 2023*, volume 13871 of *LNCS*, pages 553–572. Springer, Cham, April 2023.

[ANP20] Alexandre Adomnicăi, Zakaria Najm, and Thomas Peyrin. Fixslicing: A New GIFT Representation - Fast Constant-Time Implementations of GIFT and GIFT-COFB on ARM Cortex-M. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2020(3):402–427, Jun. 2020.

[AP21] Alexandre Adomnicăi and Thomas Peyrin. Fixslicing AES-like Ciphers: New bitsliced AES speed records on ARM-Cortex M and RISC-V. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2021(1):402–425, 2021.

[BA20] Elnaz Bagherzadeh and Zahra Ahmadian. Milp-based automatic differential search for LEA and HIGHT block ciphers. *IET Inf. Secur.*, 14(5):595–603, 2020.

[BCD+19] Zhenzhen Bao, Avik Chakraborti, Nilanjan Datta, Jian Guo, Mridul Nandi, Thomas Peyrin, and Kan Yasuda. Photon-Beetle. A submission to NIST Lightweight Cryptography, 2019.

[BCI+19] Subhadeep Banik, Avik Chakraborti, Akiko Inoue, Tetsu Iwata, Kazuhiko Minematsu, Mridul Nandi, Thomas Peyrin, Yu Sasaki, Siang Meng Sim, and Yosuke Todo. GIFT-COFB. A submission to NIST Lightweight Cryptography, 2019.

[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, *EUROCRYPT 2023, Part IV*, volume 14007 of *LNCS*, pages 532–561. Springer, Cham, April 2023.

[BJL+14] Federico Boccardi, Robert W. Heath Jr., Angel E. Lozano, Thomas L. Marzetta, and Petar Popovski. Five disruptive technology directions for 5G. *IEEE Commun. Mag.*, 52(2):74–80, 2014.

[BN00] Mihir Bellare and Chanathip Namprempre. Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. In Tatsuaki Okamoto, editor, *ASIACRYPT 2000*, volume 1976 of *LNCS*, pages 531–545. Springer, Berlin, Heidelberg, December 2000.

[BPN+16] Carsten Bockelmann, Nuno Pratas, Hosein Nikopour, Kelvin Au, Tommy Svensson, Cedomir Stefanovic, Petar Popovski, and Armin Dekorsy. Massive machine-type communications in 5G: physical and MAC-layer solutions. *IEEE Commun. Mag.*, 54(9):59–65, 2016.

[BPP+17] Subhadeep Banik, Sumit Kumar Pandey, Thomas Peyrin, Yu Sasaki, Siang Meng Sim, and Yosuke Todo. GIFT: A Small Present. In Wieland Fischer and Naofumi Homma, editors, *Cryptographic Hardware and Embedded Systems - CHES 2017*, volume 10529 of *LNCS*, pages 321–345. Springer, 2017.

[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, *ASIACRYPT 2000*, volume 1976 of *LNCS*, pages 317–330. Springer, Berlin, Heidelberg, December 2000.

[can24] CAN Bus in Industrial Automation. https://dorleco.com/can-bus-in-industrial-automation/, 2024. Accessed: 2024-12-26.

[CBY23] Yi Chen, Zhenzhen Bao, and Hongbo Yu. Differential-linear approximation semi-unconstrained searching and partition tree: Application to LEA and speck. In Jian Guo and Ron Steinfeld, editors, *ASIACRYPT 2023, Part III*, volume 14440 of *LNCS*, pages 223–255. Springer, Singapore, December 2023.

[CDJ+21] Avik Chakraborti, Nilanjan Datta, Ashwin Jha, Cuauhtemoc Mancillas-López, Mridul Nandi, and Yu Sasaki. Elastic-tweak: A framework for short tweak tweakable block cipher. In Avishek Adhikari, Ralf Küsters, and Bart Preneel, editors, *INDOCRYPT 2021*, volume 13143 of *LNCS*, pages 114–137. Springer, Cham, December 2021.

[CDN+23] Benoît Cogliati, Avijit Dutta, Mridul Nandi, Jacques Patarin, and Abishanka Saha. Proof of mirror theory for a wide range of $\xi_{\max}$. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part IV*, volume 14007 of *LNCS*, pages 470–501. Springer, Cham, April 2023.

[CIMN17] Avik Chakraborti, Tetsu Iwata, Kazuhiko Minematsu, and Mridul Nandi. Blockcipher-based authenticated encryption: How small can we go? In Wieland Fischer and Naofumi Homma, editors, *CHES 2017*, volume 10529 of *LNCS*, pages 277–298. Springer, Cham, September 2017.

[CLL24] Wonseok Choi, Jooyoung Lee, and Yeongmin Lee. Toward full n-bit security and nonce misuse resistance of block cipher-based MACs. In Kai-Min Chung and Yu Sasaki, editors, *ASIACRYPT 2024, Part IX*, volume 15492 of *LNCS*, pages 251–279. Springer, Singapore, December 2024.

[CLLL21] Wonseok Choi, ByeongHak Lee, Jooyoung Lee, and Yeongmin Lee. Toward a fully secure authenticated encryption scheme from a pseudorandom permutation. In Mehdi Tibouchi and Huaxiong Wang, editors, *ASIACRYPT 2021, Part III*, volume 13092 of *LNCS*, pages 407–434. Springer, Cham, December 2021.

[CS14] Shan Chen and John P. Steinberger. Tight security bounds for key-alternating ciphers. In Phong Q. Nguyen and Elisabeth Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 327–350. Springer, Berlin, Heidelberg, May 2014.

[CS16] Benoît Cogliati and Yannick Seurin. EWCDM: An efficient, beyond-birthday secure, nonce-misuse resistant MAC. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part I*, volume 9814 of *LNCS*, pages 121–149. Springer, Berlin, Heidelberg, August 2016.

[Dav24] Nahla Davies. The Encryption Enigma: Securing Automated Processes, 2024. Accessed: 2024-12-29.

[DEMS19] Christoph Dobraunig, Maria Eichlseder, Florian Mendel, and Martin Schläffer. Ascon. A submission to NIST Lightweight Cryptography, 2019.

[DHP+19] Joan Daemen, Seth Hoffert, Michaël Peeters, Gilles Van Assche, Ronny Van Keer, and Silvia Mella. Xoodyak. A submission to NIST Lightweight Cryptography, 2019.

[DHT17] Wei Dai, Viet Tung Hoang, and Stefano Tessaro. Information-theoretic indistinguishability via the chi-squared method. In Jonathan Katz and Hovav Shacham, editors, *CRYPTO 2017, Part III*, volume 10403 of *LNCS*, pages 497–523. Springer, Cham, August 2017.

[DKP16] Giuseppe Durisi, Tobias Koch, and Petar Popovski. Toward Massive, Ultra-reliable, and Low-Latency Wireless Communication With Short Packets. *Proc. IEEE*, 104(9):1711–1726, 2016.

[DLS18] Giuseppe Durisi, Gianluigi Liva, and Fabian Steiner. Short-packet communications: Fundamentals and practical coding schemes. IEEE Globecom Tutorial, 2018.

[DNS20] Avijit Dutta, Mridul Nandi, and Abishanka Saha. Proof of Mirror Theory for $\xi_{\max} = 2$. IACR Cryptology ePrint Archive, Report 2020/669, 2020.

[DS18] Ashutosh Dhar Dwivedi and Gautam Srivastava. Differential cryptanalysis of round-reduced LEA. *IEEE Access*, 6:79105–79113, 2018.

[Dwo07] Morris Dworkin. Recommendation for Block Cipher Modes of Operation: the CCM Mode for Authentication and Confidentiality. *NIST Special Publication*, 800-38C, 2007.

[eno20] EnOcean Serial Protocol 3 (ESP3) Specification, 2020. Accessed: 2024-10-14.

[Fer20] Loïc Ferreira. (In)security of the Radio Interface in Sigfox. Cryptology ePrint Archive, Paper 2020/1575, 2020.

[GGM18] Shoni Gilboa, Shay Gueron, and Ben Morris. How Many Queries are Needed to Distinguish a Truncated Random Permutation from a Random Function? *Journal of Cryptology*, 31(1):162–171, 2018.

[GIK+19] Chun Guo, Tetsu Iwata, Mustafa Khairallah, Kazuhiko Minematsu, and Thomas Peyrin. Romulus. A submission to NIST Lightweight Cryptography, 2019.

[GL12] Conrado Porto Lopes Gouvêa and Julio Cesar López-Hernández. High speed implementation of authenticated encryption for the MSP430X microcontroller. In Alejandro Hevia and Gregory Neven, editors, *LATINCRYPT 2012*, volume 7533 of *LNCS*, pages 288–304. Springer, Berlin, Heidelberg, October 2012.

[GLL19] Shay Gueron, Adam Langley, and Yehuda Lindell. RFC 8452: AES-GCM-SIV: Nonce Misuse-Resistant Authenticated Encryption, 2019.

[GMHT22] Martin Gunnarsson, Krzysztof Mateusz Malarski, Rikard Höglund, and Marco Tiloca. Performance evaluation of group OSCORE for secure group communication in the internet of things. *ACM Trans. Internet Things*, 3(3):19:1–19:31, 2022.

[GTW24] Felix Günther, Martin Thomson, and Christopher A. Wood. Usage Limits on AEAD Algorithms. Internet-Draft draft-irtf-cfrg-aead-limits-09, Internet Engineering Task Force, October 2024. Work in Progress.

[HLK+14] Deukjo Hong, Jung-Keun Lee, Dong-Chan Kim, Daesung Kwon, Kwon Ho Ryu, and Dong-Geon Lee. LEA: A 128-bit block cipher for fast encryption on common processors. In Yongdae Kim, Heejo Lee, and Adrian Perrig, editors, *WISA 13*, volume 8267 of *LNCS*, pages 3–27. Springer, Cham, August 2014.

[HT24] Rikard Höglund and Marco Tiloca. Key Usage Limits for OSCORE. Internet-Draft draft-ietf-core-oscore-key-limits-03, Internet Engineering Task Force, July 2024. Work in Progress.

[IMGM15] Tetsu Iwata, Kazuhiko Minematsu, Jian Guo, and Sumio Morioka. CLOC: Authenticated encryption for short input. In Carlos Cid and Christian Rechberger, editors, *FSE 2014*, volume 8540 of *LNCS*, pages 149–167. Springer, Berlin, Heidelberg, March 2015.

[IS17] Tetsu Iwata and Yannick Seurin. Reconsidering the security bound of AES-GCM-SIV. *IACR Trans. Symm. Cryptol.*, 2017(4):240–267, 2017.

[ISO19] ISO/IEC 29192-2:2019, Information security – Lightweight cryptography – Part 2: Block ciphers., 2019.

[Kho14] Dmitry Khovratovich. Key wrapping with a fixed permutation. In Josh Benaloh, editor, *CT-RSA 2014*, volume 8366 of *LNCS*, pages 481–499. Springer, Cham, February 2014.

[KKS20] Dongyeong Kim, Dawoon Kwon, and Junghwan Song. Efficient computation of boomerang connection probability for arx-based block ciphers with application to SPECK and LEA. *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.*, 103-A(4):677–685, 2020.

[KR11] Ted Krovetz and Phillip Rogaway. The software performance of authenticated-encryption modes. In Antoine Joux, editor, *FSE 2011*, volume 6733 of *LNCS*, pages 306–327. Springer, Berlin, Heidelberg, February 2011.

[KSX16] 128-bit block cipher LEA. KS X 3246 (in Korean), 2016.

[KY01] Jonathan Katz and Moti Yung. Unforgeable encryption and chosen ciphertext secure modes of operation. In Bruce Schneier, editor, *FSE 2000*, volume 1978 of *LNCS*, pages 284–299. Springer, Berlin, Heidelberg, April 2001.

[Luc00] Stefan Lucks. The sum of PRPs is a secure PRF. In Bart Preneel, editor, *EUROCRYPT 2000*, volume 1807 of *LNCS*, pages 470–484. Springer, Berlin, Heidelberg, May 2000.

[MAMK18] Hassan Malik, Muhammad Mahtab Alam, Yannick Le Moullec, and Alar Kuusik. NarrowBand-IoT Performance Analysis for Healthcare Applications. In *ANT/SEIT*, volume 130 of *Procedia Computer Science*, pages 1077–1083. Elsevier, 2018.

[Mor07] Morris J. Dworkin. Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC. *NIST Special Publication*, 800-38D, Nov 28 2007.

[nis07a] Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC. NIST Special Publication 800-38D, 2007. National Institute of Standards and Technology.

[nis07b] Recommendation for Block Cipher Modes of Operation: the CCM Mode for Authentication and Confidentiality. NIST Special Publication 800-38C, 2007. National Institute of Standards and Technology.

[NOMI15] Yuichi Niwa, Keisuke Ohashi, Kazuhiko Minematsu, and Tetsu Iwata. GCM security bounds reconsidered. In Gregor Leander, editor, *FSE 2015*, volume 9054 of *LNCS*, pages 385–407. Springer, Berlin, Heidelberg, March 2015.

[NSS20] Yusuke Naito, Yu Sasaki, and Takeshi Sugawara. LM-DAE: low-memory deterministic authenticated encryption for 128-bit security. *IACR Trans. Symmetric Cryptol.*, 2020(4):1–38, 2020.

[NSS21] Yusuke Naito, Yu Sasaki, and Takeshi Sugawara. Double-block-length hash function for minimum memory size. In Mehdi Tibouchi and Huaxiong Wang, editors, *ASIACRYPT*, volume 13092 of *Lecture Notes in Computer Science*, pages 376–406. Springer, 2021.

[Pat90] Jacques Patarin. Pseudorandom permutations based on the des scheme. In *International Symposium on Coding Theory and Applications (held in Europe)*, pages 193–204. Springer, 1990.

[PBB+17] John Padgette, John Bahr, Mayank Batra, Marcel Holtmann, Rhonda Smithbey, Lily Chen, and Karen Scarfone. Guide to Bluetooth Security. *NIST Special Publication*, 800-121, May 17 2017.

[Poe07] Bertram Poettering. AVRAES: The AES block cipher on AVR controllers. http://point-at-infinity.org/avraes/, 2007. Accessed: 2024-09-12.

[qrc24] ISO/IEC 18004:2024 Information technology - Automatic identification and data capture techniques - QR code bar code symbology specification, 2024.

[Rog02] Phillip Rogaway. Authenticated-encryption with associated-data. In Vijayalakshmi Atluri, editor, *ACM CCS 2002*, pages 98–107. ACM Press, November 2002.

[SHB14] Zach Shelby, Klaus Hartke, and Carsten Bormann. The Constrained Application Protocol (CoAP). RFC 7252, June 2014.

[Sho96] Victor Shoup. On fast and provably secure message authentication based on universal hashing. In Neal Koblitz, editor, *Advances in Cryptology — CRYPTO '96*, pages 313–328, Berlin, Heidelberg, 1996. Springer Berlin Heidelberg.

[SHY16] Ling Song, Zhangjie Huang, and Qianqian Yang. Automatic differential analysis of ARX block ciphers with application to SPECK and LEA. In Joseph K. Liu and Ron Steinfeld, editors, *ACISP 16, Part II*, volume 9723 of *LNCS*, pages 379–394. Springer, Cham, July 2016.

[sig17] Sigfox Technical Overview, 2017. Accessed: 2024-10-14.

[SMPS19] Göran Selander, John Preuß Mattsson, Francesca Palombini, and Ludwig Seitz. Object Security for Constrained RESTful Environments (OSCORE). RFC 8613, July 2019.

[SS23] Yaobin Shen and François-Xavier Standaert. Optimally secure tweakable block ciphers with a large tweak from n-bit block ciphers. *IACR Trans. Symmetric Cryptol.*, 2023(2):47–68, 2023.

[SWW17] Ling Sun, Wei Wang, and Meiqin Wang. Automatic search of bit-based division property for ARX ciphers and word-based division property. In Tsuyoshi Takagi and Thomas Peyrin, editors, *ASIACRYPT 2017, Part I*, volume 10624 of *LNCS*, pages 128–157. Springer, Cham, December 2017.

[SWW21] Ling Sun, Wei Wang, and Meiqin Wang. Linear cryptanalyses of three AEADs with GIFT-128 as underlying primitives. *IACR Trans. Symm. Cryptol.*, 2021(2):199–221, 2021.

[SWW22] Ling Sun, Wei Wang, and Meiqin Wang. Addendum to linear cryptanalyses of three AEADs with GIFT-128 as underlying primitives. *IACR Trans. Symm. Cryptol.*, 2022(1):212–219, 2022.

[TMC+24] Meltem Sönmez Turan, Kerry McKay, Donghoon Chang, Jinkeon Kang, and John Kelsey. Ascon-Based Lightweight Cryptography Standards for Constrained Devices: Authenticated Encryption, Hash, and Extendable Output Functions. NIST SP 800-232 (Initial Public Draft), 2024. National Institute of Standards and Technology.

[WLHL24] Shichang Wang, Meicheng Liu, Shiqi Hou, and Dongdai Lin. Differential-linear cryptanalysis of GIFT family and GIFT-based ciphers. *IACR Communications in Cryptology*, 1(1), 2024.

[ZDC+21] Rui Zong, Xiaoyang Dong, Huaifeng Chen, Yiyuan Luo, Si Wang, and Zheng Li. Towards key-recovery-attack friendly distinguishers: Application to GIFT-128. *IACR Trans. Symm. Cryptol.*, 2021(1):156–184, 2021.

[ZGH16] Kai Zhang, Jie Guan, and Bin Hu. Zero correlation linear cryptanalysis on LEA family ciphers. *J. Commun.*, 11(7):677–685, 2016.

A Mirror Theory

The goal of the mirror theory is to compute a lower bound of the number of solutions to a certain type of system of equations and inequalities.

We consider a system of equations and inequalities $\gamma = (\gamma^{=}, \gamma^{\neq})$, which is divided into a system of equations $\gamma^{=}$ and a system of inequalities $\gamma^{\neq}$. A set of variables $\mathcal{V}$ is partitioned into $\mathcal{V}_1 \sqcup \mathcal{V}_2$. Intuitively, variables in $\mathcal{V}_1$ come from one permutation and ones in $\mathcal{V}_2$ are results of the other permutation. In this section, we assume that they are arbitrarily partitioned. So, the variables in $\mathcal{V}_1$ (or $\mathcal{V}_2$) should be distinct. We use the notion $X \sim Y$ to indicate that X and Y belong to the same subset meaning that X and Y are distinct elements within that subset. Additionally, we impose the following constraint on γ: If $X \sim Y$, then $X \neq Y$.

Fix a positive integer c. For $1 \leq i \leq c$ and a positive integer $\xi_i > 1$, the system of equations as $\gamma^{=}$ is represented as:

$$\gamma^{=} : \begin{cases} X_{1,0} \oplus X_{1,1} = \lambda_{1,1}, \ldots, X_{1,0} \oplus X_{1,\xi_1-1} = \lambda_{1,\xi_1-1}, \\ \qquad\qquad \vdots \\ X_{c,0} \oplus X_{c,1} = \lambda_{c,1}, \ldots, X_{c,0} \oplus X_{c,\xi_c-1} = \lambda_{c,\xi_c-1} \end{cases}$$

where $\lambda_{\alpha,i} \in \{0,1\}^n$ for $1 \leq \alpha \leq c$ and $0 \leq i \leq \xi_\alpha - 1$. The set of variables on $\gamma^{=}$ is denoted as $\mathcal{V}^{=}$ and we define $\mathcal{V}_1^{=} \coloneqq \mathcal{V}^{=} \cap \mathcal{V}_1$ and $\mathcal{V}_2^{=} \coloneqq \mathcal{V}^{=} \cap \mathcal{V}_2$. We also define $\mathcal{V}^{\neq} \coloneqq \mathcal{V} \setminus \mathcal{V}^{=}$, $\mathcal{V}_1^{\neq} \coloneqq \mathcal{V}^{\neq} \cap \mathcal{V}_1$ and $\mathcal{V}_2^{\neq} \coloneqq \mathcal{V}^{\neq} \cap \mathcal{V}_2$. The set of variables $\mathcal{V}^{=}$ consists of c components, and for $i \in [1..c]$, the i-th component takes form of $\{X_{i,0}, \ldots, X_{i,\xi_i-1}\}$. The largest number of components is denoted as $\xi_{\max}$, where $\xi_{\max} = \max_{i \in [1..c]} \{\xi_i\}$.

We separately establish a system of inequalities with $\gamma^{=}$. For a non-negative integer v, we denote

$$\gamma^{\neq} : \begin{cases} X'_1 \oplus X'_2 \neq \lambda'_1, \\ X'_3 \oplus X'_4 \neq \lambda'_2, \\ \qquad \vdots \\ X'_{2v-1} \oplus X'_{2v} \neq \lambda'_v \end{cases}$$

where $\lambda'_i \in \{0,1\}^n$ for $1 \leq i \leq v$. Note that a variable that appears in both systems of equations and inequalities can be represented by a single symbol. However, to clearly distinguish between the system of equations and the system of inequalities, we use separate symbols for those in $\gamma^{=}$ and those in $\gamma^{\neq}$. The equivalence between variables is indicated using the relation $\sim_{\mathsf{eq}}$. Specifically, for some i, X'_i can be identified as an element of $\mathcal{V}^{=}$ or another element of $\mathcal{V}^{\neq}$. This identification is publicly known and can be denoted as a relation $\sim_{\mathsf{eq}}$, i.e., $X'_i \sim_{\mathsf{eq}} X_{j,k} \Leftrightarrow X'_i = X_{j,k}$ and $X'_i \sim_{\mathsf{eq}} X'_j \Leftrightarrow X'_i = X'_j$.

In this paper, we express the system of equations and inequalities with relation $\sim$ and $\sim_{\mathsf{eq}}$; denoted as $\Gamma \coloneqq (\gamma^{=}, \gamma^{\neq}, \sim, \sim_{\mathsf{eq}})$. $h(\Gamma)$ denotes the number of solutions to γ subject to the above constraints.

In this work, we focus on a system Γ for which at least one solution exists. To ensure a solution, the system must satisfy the non-degeneracy properties outlined below:

1 $\lambda_{\alpha,i} \neq 0$ for all $\alpha \in [1..c]$ and $i \in [1..\xi_\alpha - 1]$ such that $X_{\alpha,0} \sim X_{\alpha,i}$.

2 $\lambda_{\alpha,i} \neq \lambda_{\alpha,j}$ for all $\alpha \in [1..c]$ and distinct $i, j \in [1..\xi_\alpha - 1]$ such that $X_{\alpha,i} \sim X_{\alpha,j}$.

3 There is no (α, β, i, j) such that $\gamma^{=}$ contains $X_{\alpha,i} \oplus X_{\alpha,j} = \lambda'_\beta$ and $\gamma^{\neq}$ contains $X_{\alpha,i} \oplus X_{\alpha,j} \neq \lambda'_\beta$.

We refer to any system Γ satisfying the above properties as a *nice* system. The following theorem provides a lower bound of $h(\Gamma)$ for a nice system Γ, which will be used in our proofs.

Theorem 5. *Let Γ be a nice system over $\{0,1\}^n$ such that the number of equations is q and the number of inequalities is v. Suppose the number of variables in the largest component of $\gamma^=$ is $\xi_{\max}$. If $\xi_{\max}^2 n + \xi_{\max} \le 2^{n/2}$, $q\xi_{\max}^2 \le \frac{2^n}{12}$ and $q + v \le 2^{n-1}$, one has*

$$h(\Gamma) \ge \frac{(2^n-2)_{|\mathcal{V}_1|}(2^n-2)_{|\mathcal{V}_2|}}{2^{nq}} \left(1 - \frac{2v}{2^n}\right).$$

The proof of Theorem 5 is referred to Choi, Lee and Lee [CLL24].

B Full Proof of Theorem 4: Birthday Attack on GAE with $t < n$

Without loss of generality, assume that $c_1 = 0$, $c_2 = 0$, $t_1 = 0$, $t_2 = 0$, $\mathrm{Const}_3 = 0^n$, and $\mathrm{Const}_4 = 0^t$, since the constants, nonces, and plaintexts are public for the adversary and do not affect the security of GAE.

If $\mathrm{rank}(\mathsf{Fin}) \le 1$, then as the proof of Theorem 3, one can construct a $(1,0,O(1))$-adversary $\mathcal{A}$ such that $\mathbf{Adv}^{\mathsf{nAE}}_{\mathsf{GAE}[E_{K_1},E_{K_2}]}(\mathcal{A}) = O(1)$. We thus assume that $\mathrm{rank}(\mathsf{Fin}) = 2$. In the following evaluation, we use the following conditions in the real world.

$$\text{If } c_4 \neq 0, \text{ then } W_i = W_j \Leftrightarrow (t_4 \cdot c_4^{-1} \cdot c_3) \cdot W_i = (t_4 \cdot c_4^{-1} \cdot c_3) \cdot W_j$$
$$\Rightarrow \mathsf{C}_6 : \mathsf{lsb}_t(t_4 \cdot c_4^{-1} \cdot C_i) \oplus T_i = \mathsf{lsb}_t(t_4 \cdot c_4^{-1} \cdot C_j) \oplus T_j. \quad (6)$$
$$\text{If } c_4 = 0 \wedge c_3 \neq 0, \text{ then } W_i = W_j \Leftrightarrow \mathsf{C}_7 : C_i = C_j. \quad (7)$$
$$\text{If } c_3 \neq 0, \text{ then } Y_i = Y_j \Leftrightarrow (t_3 \cdot c_3^{-1} \cdot c_4) \cdot Y_i = (t_3 \cdot c_3^{-1} \cdot c_4) \cdot Y_j$$
$$\Rightarrow \mathsf{C}_8 : \mathsf{lsb}_t(t_3 \cdot c_3^{-1} \cdot C_i) \oplus T_i = \mathsf{lsb}_t(t_3 \cdot c_3^{-1} \cdot C_j) \oplus T_j. \quad (8)$$
$$\text{If } c_3 = 0 \wedge c_4 \neq 0, \text{ then } Y_i = Y_j \Leftrightarrow \mathsf{C}_9 : C_i = C_j. \quad (9)$$

We then give adversaries on GAE with the following cases.

B.1 $v_1 = 0$

Fixing a plaintext M, for any pair of nonces (N_1, N_2), the ciphertext blocks of E_{K_1} are the same, i.e., $W_1 = W_2$, which can be confirmed by the condition $(\mathsf{C}_6 \wedge c_4 \neq 0) \vee (\mathsf{C}_7 \wedge c_4 = 0)$. In the ideal world, we have $\Pr[(\mathsf{C}_6 \wedge c_4 \neq 0) \vee (\mathsf{C}_7 \wedge c_4 = 0)] \le \frac{1}{2^t}$. With the condition, we can construct a $(2,0,O(1))$-adversary $\mathcal{A}$ such that $\mathbf{Adv}^{\mathsf{nAE}}_{\mathsf{GAE}[E_{K_1},E_{K_2}]}(\mathcal{A}) = O(1)$. The adversary is given in the following.

1 Choose a plaintext M and distinct nonces N_1, N_2.

2 For $i \in [2]$, make an encryption query (N_i, M) and receive the pair (C_i, T_i).

3 If $(\mathsf{C}_6 \wedge c_4 \neq 0) \vee (\mathsf{C}_7 \wedge c_4 = 0)$, then return 1; otherwise return 0.

In the real world, the adversary returns 1. In the ideal world, it returns 0 with the probability of almost 1.

B.2 $v_1 \neq 0 \wedge v_2 \neq 0 \wedge \left(\exists (N_1, M_1), (N_2, M_2) \text{ s.t. } (N_1, M_1) \neq (N_2, M_2) \wedge V_1 = V_2\right)$

With the distinct pairs (N_1, M_1) and (N_2, M_2) such that $V_1 = V_2$, one has the collision $W_1 = W_2$, which can be confirmed by $(\mathsf{C}_6 \wedge c_4 \neq 0) \vee (\mathsf{C}_7 \wedge c_4 = 0)$ in the real world. In the ideal world, we have $\Pr[(\mathsf{C}_6 \wedge c_4 \neq 0) \vee (\mathsf{C}_7 \wedge c_4 = 0)] \le \frac{1}{2^t}$. With the equation, we can construct a $(2,0,O(1))$-adversary $\mathcal{A}$ such that $\mathbf{Adv}^{\mathsf{nAE}}_{\mathsf{GAE}[E_{K_1},E_{K_2}]}(\mathcal{A}) = O(1)$. The adversary is given in the following.

1 Choose pairs $(N_1, M_1), (N_2, M_2)$ such that $(N_1, M_1) \neq (N_2, M_2) \wedge V_1 = V_2$.

2 For $i \in \{1, 2\}$, make an encryption query (N_i, M_i) and receive the pair (C_i, T_i).

3 If $(\mathsf{C}_6 \wedge c_4 \neq 0) \vee (\mathsf{C}_7 \wedge c_4 = 0)$, then return 1; otherwise return 0.

In the real world, the adversary returns 1. In the ideal world, it returns 0 with a probability of almost 1.

B.3 $v_1 \neq 0 \wedge v_2 \neq 0 \wedge \left(\forall (N_1, M_1), (N_2, M_2) \text{ s.t. } (N_1, M_1) \neq (N_2, M_2) : V_1 \neq V_2\right)$

In this case, some bits of an n-bit plaintext are not used to define the plaintext block V, and thus $x_2 \neq 0$ or $c_2 \neq 0$ must hold to encrypt the part of the plaintext. If $x_2 \neq 0$, then by $t < n$, one cannot recover W or Y, and the decryption cannot be performed since it requires the plaintext. If $x_2 = 0$, then some bits of a plaintext are not used to define the tag. With the bits, one can construct a $(1, 1, O(1))$-adversary $\mathcal{A}$ such that $\mathbf{Adv}^{\mathsf{nAE}}_{\mathsf{GAE}[E_{K_1}, E_{K_2}]}(\mathcal{A}) = O(1)$. The adversary is defined in the following.

1 Choose nonce N and a plaintext N, make the encryption query (N, M), and receive the response (C, T).

2 Make a decryption query $(N, C \oplus \Delta, T)$ such that $\Delta \in \{0, 1\}^n$ is not used to define the tag.

3 If $\perp$ is returned, then return 0, and otherwise return 1.

In the real world, the adversary returns 1. In the ideal world, it returns 0 with a probability of almost 1.

B.4 $v_1 \neq 0 \wedge v_2 = 0 \wedge x_2 = 0$

In this case, a plaintext M is not used in defining the plaintext blocks V and X. Hence, one can construct a $(1, 1, O(1))$-adversary $\mathcal{A}$ forging a tag such that $\mathbf{Adv}^{\mathsf{nAE}}_{\mathsf{GAE}[E_{K_1}, E_{K_2}]}(\mathcal{A}) = O(1)$. The adversary is defined in the following.

1 Choose nonce N and a plaintext N, make the encryption query (N, M), and receive the response (C, T).

2 Make a decryption query $(N, C \oplus 1^n, T)$.

3 If $\perp$ is returned, then return 0, and otherwise return 1.

In the real world, the adversary returns 1. In the ideal world, it returns 0.

B.5 $v_1 \neq 0 \wedge v_2 = 0 \wedge x_2 \neq 0 \wedge x_3 = 0$

In the real world, one can define different pairs of nonce and plaintext (N_1, M_1) and (N_2, M_2) such that $X_1 = X_2$ ($\Leftrightarrow Y_1 = Y_2$). The equation $Y_1 = Y_2$ can be confirmed by the condition $(\mathsf{C}_8 \wedge c_3 \neq 0^n) \vee (\mathsf{C}_9 \wedge c_3 = 0^n)$. With the pairs and the condition, we can construct a $(2, 0, O(1))$-adversary such that $\mathbf{Adv}^{\mathsf{nAE}}_{\mathsf{GAE}[E_{K_1}, E_{K_2}]}(2, 0, O(1)) = O(1)$. The adversary is defined in the following.

1 Choose pairs $(N_1, M_1), (N_2, M_2)$ such that $(N_1, M_1) \neq (N_2, M_2) \wedge X_1 = X_2\big)$.

2 For $i \in \{1, 2\}$, make an encryption query (N_i, M_i) and receive the pair (C_i, T_i).

3 If $(\mathsf{C}_8 \wedge c_3 \neq 0) \vee (\mathsf{C}_9 \wedge c_3 = 0)$, then return 1; otherwise return 0.

In the real world, the adversary returns 1. In the ideal world, it returns 0 with the probability of almost 1.

B.6 $v_1 \neq 0 \wedge v_2 = 0 \wedge x_2 \neq 0 \wedge x_3 \neq 0$

We define an adversary in the following.

1 Choose q_e pairs of nonce and plaintext $(N_1, M_1), \ldots, (N_{q_e}, M_{q_e})$ such that the nonces are all distinct and the plaintexts are respectively chosen uniformly at random from $\{0,1\}^n$.

2 For each $i \in [1..q_e]$, make an encryption query (N_i, M_i) and receive the pair (C_i, T_i).

3 For each pair $(i,j) \in [1..q_e]^2$ s.t. $i \neq j$, if $(\mathsf{C}_8 \wedge c_3 \neq 0) \vee (\mathsf{C}_9 \wedge c_3 = 0)$ (i.e., $Y_i = Y_j$ holds with high probability), then return 1.

4 Return 0.

In the real world, for $(i,j) \in [1..q_e]^2$ s.t. $i \neq j$, there are two cases for the condition $\mathsf{C}_{8,9} := (\mathsf{C}_8 \wedge c_3 \neq 0) \vee (\mathsf{C}_9 \wedge c_3 = 0)$: (A) $V_i = V_j \wedge \mathsf{C}_{8,9}$; (B) $V_i \neq V_j \wedge \mathsf{C}_{8,9}$. We then have $\Pr[V_i = V_j \wedge \mathsf{C}_{8,9}] = \Pr[V_i = V_j] \geq \frac{1}{2^n}$ and $\Pr[V_i \neq V_j \wedge \mathsf{C}_{8,9}] \geq \left(1 - \frac{1}{2^n}\right) \cdot \frac{1}{2^t}$. We thus have $\Pr\left[\exists i \neq j \in [1..q_e]^2 \text{ s.t. } \mathsf{C}_{8,9}\right] \geq \binom{q_e}{2} \cdot \left(\frac{1}{2^t} + \frac{1}{2^n} - \frac{1}{2^{t+n}}\right)$. In the ideal world, we have $\Pr[\exists i \neq j \in [1..q_e] \text{ s.t. } \mathsf{C}_{8,9}] \leq \binom{q_e}{2} \cdot \frac{1}{2^t}$.

By these bounds, there exists a $(q_e, 0, O(q_e^2))$-adversary $\mathcal{A}$ such that

$$\mathbf{Adv}^{\mathsf{nAE}}_{\mathsf{GAE}[E_{K_1}, E_{K_2}]}(\mathcal{A}) = O\left(\binom{q_e}{2} \cdot \frac{1}{2^n}\right) = O\left(\frac{q_e^2}{2^n}\right).$$