Rugged Pseudorandom Permutations with Beyond-Birthday-Bound Security

Nilanjan Datta[1,3], Jean Paul Degabriele[2], Avijit Dutta[1,3], Vukašin Karadžić[4], and Hrithik Nandi[1,5]

[1] Institute for Advancing Intelligence (IAI) TCG CREST, Kolkata, India
[2] Technology Innovation Institute, Abu Dhabi, UAE
[3] Academy of Scientific and Innovative Research (AcSIR), Ghaziabad, India
[4] Technische Universität Darmstadt, Germany
[5] Ramakrishna Mission Vivekananda Educational and Research Institute, India
nilanjan.datta@tcgcrest.org, jeanpaul.degabriele@tii.ae, avirocks.dutta13@gmail.com, vukasin.karadzic@tu-darmstadt.de, hrithik.nandi.85@tcgcrest.org

Abstract. A rugged pseudorandom permutation (RPRP) is a security notion for variable-length tweakable ciphers that is strictly weaker than the traditional notion of a strong pseudorandom permutation. Being a weaker security notion it admits more efficient constructions. Yet the notion is strong enough so that any such construction can lend itself to a number of practical applications. It can be used to construct onion encryption, misuse-resistant AEAD, and AEAD secure under the release of unverified plaintext. Two recent works have introduced the notion, explored some of its applications, and studied a number of constructions that meet this notion. However, no constructions are known to achieve this notion with beyond-birthday-bound security. Current cloud applications are processing amounts of data that go well beyond the traditional 2^{32} barrier, and 2^{64} is becoming the new target. As such, the need for encryption with beyond-birthday-bound security has become a very practical concern. In this work, we present the first constructions for variable-length tweakable ciphers that satisfy RPRP security beyond the birthday bound. From these constructions, we readily obtain efficient AEAD schemes that are optimally secure against nonce misuse and the release of unverified plaintext.

Keywords: Beyond Birthday Bound Security, Rugged Pseudorandom Permutation, AEAD, Misuse Resistance, Release of Unverified Plaintext.

1 Introduction

Authenticated encryption with stronger security properties has been the subject of several research works [3, 6, 7, 27, 40], but there has been rather limited deployment of such AEAD schemes in practice. There are a number of possible reasons for this, such as: practitioners not valuing or not understanding these stronger security properties, practitioners not willing to pay the performance penalty that such schemes incur, and, perhaps most importantly, the lack of standardisation of such schemes. Indeed, for many companies, standardisation is often a necessary requirement before adopting and implementing a new cryptographic scheme. However, NIST is currently aiming to remedy this by standardising an Accordion mode cipher [35]. An Accordion mode is a tweakable variable-length cipher construction that is secure as a strong pseudorandom permutation. One of the central applications of an Accordion mode cipher is that it can be readily converted into an AEAD scheme with strong security properties by applying some simple encoding on the inputs. This approach, known as the *encode-then-encipher* paradigm, was originally put forth by Bellare and Rogaway in 2000 [10], and was recently revisited, with a more modern take, by Shrimpton and Terashima in [41].

An AEAD scheme obtained via the encode-then-encipher paradigm using a Strong Pseudorandom Permutation (SPRP) is automatically resistant to the nonce misuse (MRAE) [41], and is also secure against the release of unverified plaintext (RUPAE) [20]. Informally, a misuse-resistant AEAD scheme [40] retains the best possible security when the nonce input repeats, and an AEAD scheme that is secure under the release of unverified plaintext [3, 7] achieves a strong form of non-malleability such that even if the invalid plaintext obtained from decrypting a malicious ciphertext is released to the adversary, AEAD security is still preserved. Such an AEAD scheme offers very robust security, but this added security typically incurs a performance penalty due to the extra processing involved in most constructions and the inherent requirement that both encryption and decryption necessitate two passes over their inputs. Generally speaking, the processing overhead affects the scheme's throughput, whereas the inherent two-pass nature of the constituent algorithms affects their latency and memory requirements. In addition, if encryption and decryption are two-pass, it means that they cannot process their inputs as a stream [28].

The encode-then-encipher paradigm was recently revisited by Degabriele and Karadžić, who showed that this design framework could be instantiated using a variable-length tweakable cipher satisfying a weaker security notion [20]. They called this security notion, and any variable-length tweakable cipher that satisfies it, a Rugged Pseudorandom Permutation (RPRP). An RPRP is a strictly weaker notion than an SPRP, and any SPRP construction is automatically an RPRP. By replacing an SPRP with an RPRP in the encode-then-encipher paradigm, one obtains a different trade-off between security and performance. Namely, the resulting AEAD will be, in addition to standard AEAD security, either MRAE or RUPAE secure, but not both simultaneously–as is the case with an SPRP. The RPRP notion introduces an asymmetry between the security offered by the enciphering and deciphering algorithms, where enciphering is fully non-malleable but deciphering allows some malleability. Accordingly, Degabriele and Karadžić showed that using the enciphering algorithm to encrypt via the EtE transform yields MRAE security, whereas using the deciphering algorithm to encrypt via the EtD transform yields RUPAE security [20]. Both the EtE and EtD transforms have variants which yield either nonce-based or nonce-hiding AEAD [9], and only require some lightweight encoding[6]. However, by using an RPRP instead of an SPRP, they obtain faster AEAD constructions. For instance, adapting the classical result of Luby and Rackoff to the tweakable setting [33], it is well known that an SPRP requires four rounds of Feistel, but an RPRP is readily achieved via three rounds of Feistel [21]. Two other RPRP constructions, called UIV and HEC [20,21], are obtained respectively by slashing the last layer from the SPRP constructions PIV and HCTR [12,41], each of which require three layers of processing. As a result, RPRP constructions can achieve higher throughput than SPRP constructions. In addition, RPRPs can also be advantageous in terms of latency since only the enciphering algorithm needs two passes over the data. In contrast, the deciphering algorithm only needs one pass, as is the case for the UIV and HEC constructions. Thus, the resulting MRAE-secure AEAD constructions can have online decryption, and the RUP-secure AEAD constructions can have online encryption.

In light of the above, it is more appropriate to compare an RPRP construction to a misuse-resistant or RUP-secure AEAD scheme rather than an SPRP construction. A RPRP-based AEAD scheme offers the benefit of code minimality because the same RPRP construction yields four different AEAD schemes with different properties, which is not possible with structurally different dedicated MRAE- and RUP-secure schemes. In this work, we present the first RPRP constructions with beyond-birthday-bound (BBB) security, and as a result, we obtain new AEAD schemes that are misuse-resistant or RUP-secure with beyond-birthday-bound security, and each can be either nonce-based or nonce-hiding. Both of our RPRP constructions are based on a tweakable block cipher (TBC) such as SKINNY or Deoxys-TBC and allow for tweaks of arbitrary size. Compared to prior RPRP constructions such as UIV and HEC [20,21], the main overhead incurred by our constructions in order to achieve BBB security is mainly due to the use of a TBC instead of a block cipher and a few extra evaluations. Two prominent AEAD constructions that are misuse resistant are Deoxys-AE2 [31] and Romulus-M [29], both of which are based on the SCT/SIV structure [38] and require ∼1.5 tweakable block cipher calls per message block. Instead, our constructions require one tweakable block cipher call per message block (or a couple more) and a universal hash evaluation over the message and tweak/associated data. This has the benefit that a universal hash is typically lighter to compute than a TBC call, and during deciphering, it can be evaluated in parallel using carryless multiplication for example. As for RUP-secure AEAD, besides constructions based on wide-block SPRPs, which are generally slower, we are not aware of other constructions that are BBB secure. Moreover, RPRPs have applications beyond AEAD, such as realising onion encryption in Tor, where it is crucial that the encryption layers be length preserving, thereby ruling out AEAD as a building block. Counter Galois Onion (CGO) [22] is a recent proposal by Degabriele *et al.* for an onion encryption scheme for Tor with improved security that uses an RPRP to encrypt each layer. Recently, the Tor project has announced its intent to adopt CGO [34]. Both of our RPRP constructions could be plugged into CGO to attain an onion encryption scheme with BBB security. While this is an important practical application of our work, it is beyond the scope of this paper and our exposition will mainly focus on AEAD as the main application of our new RPRP constructions. Below is our contribution in more detail.

1.1 Contribution

THEC*: BBB RPRP up to Limited Tweak Repetitions.* Our first construction is Tweakable HEC (THEC), which as the name implies, is a TBC adaptation of the block-cipher-based RPRP construction HEC [21]. Structurally, THEC bears a relation to the SPRP construction THCTR [24] where its last hashing layer

[6] The EtD transform requires the encoding to be collision resistant, but this is readily achieved by using injective encoding without the use of any hash functions.

has been shaved off, its TBC counter-mode component replaced by Counter-in-Tweak [2, 38], and the remaining universal hashes have been rearranged in a more efficient configuration. Due to these changes THEC does not suffer from the same limitations, in terms of its BBB security, pointed out by Andreeva *et al.* and Khairallah on THCTR [2, 32]. The main limitation of THEC is that it is BBB secure as an RPRP only if the tweak repetitions are limited. More specifically, THEC offers optimal RPRP security when the queried tweaks (across enciphering and deciphering queries) are distinct. On the other hand, if the number of tweak repetitions reaches $2^{n/2}$ times, then the overall security of the construction collapses to the birthday bound — often referred to as graceful security degradation. In the case of CGO in Tor, tweaks are highly unlikely to repeat and thus THEC already suffices to transform CGO into a BBB onion encryption scheme.

EtE[THEC]*: BBB MRAE Under Limited Decryptions.* We next show how to obtain a nonce-based or nonce-hiding AEAD scheme from THEC that is misuse resistant and yields secure channels with BBB security. The resulting AEAD schemes will not be BBB secure in the general AEAD sense where the number of decryption queries is unrestricted. However, in real-world protocols like TLS and QUIC, the number of decryption queries is limited because the protocol will only tolerate a limited number of decryption failures before the connection is torn down. Specifically, by applying the EtE transformation [20] to THEC, we derive a nonce-based or nonce-hiding BBB MRAE-secure AEAD scheme EtE[THEC] with a graceful security degradation on the nonce repetitions. In the EtE transform, nonce repetitions result in tweak repetitions, which may occur during encryption and decryption. Accordingly, nonce repetitions across encryption queries result in graceful security degradation, similarly to other BBB MRAE schemes like Deoxys-AE2 and Romulus-M. On the other hand, nonce repetitions across decryption queries are unrestricted in the AEAD security game and thus no longer BBB in that setting. Nevertheless, as explained in Subsection 7.1, this is still sufficient for the majority of Internet protocols like TLS and QUIC. In that setting EtE[THEC] significantly outmatches the security of AES-GCM-SIV and matches that of Deoxys-AE2.

DE-THEC*: Fully BBB RPRP with Domain Extension.* Our second, and main, RPRP construction DE-THEC, overcomes the limitations of THEC and achieves full BBB RPRP security without imposing any restriction on tweak repetitions. Another limitation of our first construction is the small size of its left domain. When translating an RPRP into a RUP-secure AEAD, the EtD transform incurs an inherent birthday-bound term in the size of the left domain [20]. Accordingly, while THEC can be translated into a RUP-secure AEAD scheme it will only be secure up to the birthday bound. On the other hand, the extended domain of DE-THEC will allow us to derive RUP-secure AEAD with BBB security.

EtE[DE-THEC] & EtD[DE-THEC]*: Fully BBB MRAE and RUPAE.* Now equipped with a superior RPRP construction (DE-THEC) we obtain two AEAD schemes that beat the state of the art in terms of misuse-resistance security (MRAE) and release-of-unverified-plaintext security (RUPAE). Applying the EtE transform yields a nonce-based or nonce-hiding AEAD scheme EtE[DE-THEC] with BBB security even in the misuse resistance setting, i.e., without graceful degradation. Notably, it offers much higher security than Deoxys-AE2 [31] and recently proposed constructions DENC2 [14] and eGCM-SIV [15] in the misuse-resistance setting. On the other hand, applying the EtD transform to DE-THEC yields a RUP-secure AEAD scheme EtE[DE-THEC] with full n-bit security with no additional restrictions. To the best of our knowledge, this is the first RUP-secure AEAD scheme with BBB security which attains significantly higher security than GCM-RUP and RIV. A comparison of our AEAD schemes with other state-of-the-art (SOTA) AEAD schemes is provided in Subsection 7.3.

1.2 Related Work

As already noted, our THEC scheme is inspired by the SPRP-secure scheme THCTR [24]. Accordingly, it is possible to extend THEC, which contains structural improvements in both of its layers, with a third layer, in order to recover a BBB-secure tweakable SPRP, although it would only be BBB if tweak repetitions are limited.

An analogous BBB-secure tweakable SPRP counterpart to our DE-THEC construction, with unrestricted tweak repetitions, is the HCTR+ construction by Datta et al. that appeared in a concurrent work to ours [17, 18]. Notably, our DE-THEC bears many similarities to the first two layers of HCTR+. The main difference, apart from the obvious one that DE-THEC and HCTR+ target different (i.e. weaker RPRP and stronger SPRP) security notions, lies in the inputs to the counter mode that ours and theirs scheme use.

2 Preliminaries

Notation. For a set $\mathcal{X}$, $X \leftarrow\!\$\, \mathcal{X}$ denotes that X is sampled uniformly at random from $\mathcal{X}$. We write $X \leftarrow Y$ to denote that Y is assigned in variable X. For any natural number q, $[q]$ denotes the set $\{0, \ldots, q-1\}$. We denote an empty set as $\emptyset$. We say two sets $\mathcal{X}$ and $\mathcal{Y}$ are disjoint if $\mathcal{X} \cap \mathcal{Y} = \emptyset$. We denote their union as $\mathcal{X} \sqcup \mathcal{Y}$ (which we refer to as *disjoint union*). For any natural number n, $\{0,1\}^n$ (resp. $\{0,1\}^{\geq n}$) denotes the set of all bit strings of length n (resp. $\geq n$) and $\{0,1\}^*$ denotes the set of all bit strings of arbitrary finite length. We denote the empty bit string with ε. Sometimes, we call an element of $\{0,1\}^n$ a *block*. For integers $1 \leq b \leq a$, $(a)_b$ denotes $a(a-1)\ldots(a-b+1)$, where $(a)_0 = 1$ by convention. For any positive real number m, $\lceil m \rceil$ denotes rounding up m to the first integer greater or equal to m, and $\lfloor m \rfloor$ denotes rounding down m to the first integer smaller or equal to m. For a bit string S, $|S| \geq m$, we denote the m most significant bits of S with $\lfloor S \rfloor_m$. For a positive integer $i < 2^n$, we write $\langle i \rangle_2$ to denote the corresponding binary representation of i encoded in n-bit string. For any function $\Phi : \mathcal{X} \to \{0,1\}^{2n}$, we define $\Phi[1]$ and $\Phi[2]$ be two functions from $\mathcal{X}$ to $\{0,1\}^n$ such that for all $x \in \mathcal{X}$, $\Phi[1](x)$ denotes the leftmost n bits of the $2n$-bit output of $\Phi(x)$, and $\Phi[2](x)$ denotes the rightmost n bits of the $2n$-bit output of $\Phi(x)$.

2.1 Tweakable Block Cipher

Let n be a natural number. A *tweakable block cipher* (TBC) is a mapping $\widetilde{\mathsf{E}} : \mathcal{K} \times \mathcal{T} \times \mathcal{X} \to \mathcal{X}$ such that for all keys $k \in \mathcal{K}$ and for all tweaks $T \in \mathcal{T}$, $X \mapsto \widetilde{\mathsf{E}}(k, T, X)$ is a permutation over $\mathcal{X}$. We denote with $\mathsf{TBC}(\mathcal{K}, \mathcal{T}, \mathcal{X})$ the set of all tweakable block ciphers with key space $\mathcal{K}$, tweak space $\mathcal{T}$, and message space $\mathcal{X}$. A *tweakable permutation* is a mapping $\widetilde{\mathsf{P}} : \mathcal{T} \times \mathcal{X} \to \mathcal{X}$ such that for all $T \in \mathcal{T}$, $X \mapsto \widetilde{\mathsf{P}}(T, X)$ is a permutation over $\mathcal{X}$. We write $\mathsf{TP}(\mathcal{T}, \mathcal{X})$ to denote the set of all tweakable permutations.

Definition 1 (STPRP security). *Let $\widetilde{\mathsf{E}} \in \mathsf{TBC}(\mathcal{K}, \mathcal{T}, \mathcal{X})$ and $\mathcal{A}$ be a deterministic, adaptive adversary. The advantage of $\mathcal{A}$ in breaking the* strong tweakable pseudorandom permutation *security of $\widetilde{\mathsf{E}}$ is defined as*

$$\mathbf{Adv}^{\mathrm{STPRP}}_{\widetilde{\mathsf{E}}}(\mathcal{A}) \triangleq \left| \Pr[k \leftarrow\!\$\, \mathcal{K}, \mathcal{A}^{\widetilde{\mathsf{E}}_k, \widetilde{\mathsf{E}}_k^{-1}} = 1] - \Pr[\widetilde{\mathsf{P}} \leftarrow\!\$\, \mathsf{TP}(\mathcal{T}, \mathcal{X}), \mathcal{A}^{\widetilde{\mathsf{P}}, \widetilde{\mathsf{P}}^{-1}} = 1] \right|.$$

Typically $\mathcal{X}$ is to be regarded as $\{0,1\}^n$. In that case, we call it a *fixed-input length* tweakable cipher.

2.2 Rugged Pseudorandom Permutation

Let EE be a variable-input length tweakable cipher over a split domain $\mathcal{X}_L \times \mathcal{X}_R$ with an associated key space $\mathcal{K}$ and the tweak space $\mathcal{T}$. For such tweakable cipher EE, we define a pair of deterministic algorithms called the encipher and the decipher algorithm, where the encipher algorithm, denoted as $\mathsf{EE}.\textsc{En}$, takes a key $k \in \mathcal{K}$, a tweak $T \in \mathcal{T}$ and a pair of strings $(X_L, X_R) \in \mathcal{X}_L \times \mathcal{X}_R$ as the input and produces $(Y_L, Y_R) \in \mathcal{X}_L \times \mathcal{X}_R$ as the output. The decipher algorithm, denoted as $\mathsf{EE}.\textsc{De}$ takes a key $k \in \mathcal{K}$, a tweak $T \in \mathcal{T}$ and a pair of strings $(Y_L, Y_R) \in \mathcal{X}_L \times \mathcal{X}_R$ as the input and produces $(X_L, X_R) \in \mathcal{X}_L \times \mathcal{X}_R$ as the output. There are two restrictions on the input query to the decipher algorithm: (i) Y_L should not repeat across the decipher queries, and (ii) it should not collide with other Y_L values that have been obtained in previous encipher queries.

The rugged pseudorandom permutation (RPRP) advantage of EE is defined as a distinguishing game between a real and an ideal world. In addition to the enciphering and deciphering oracle, the adversary is provided access to a so-called guess oracle $\mathsf{EE}.\textsc{Gu}$. The guess oracle, in the real world, takes as input a key $k \in \mathcal{K}$, a tweak $T \in \mathcal{T}$, a pair of strings $(Y_L, Y_R) \in \mathcal{X}_L \times \mathcal{X}_R$ and a non-empty finite set $\boldsymbol{V} \subseteq \mathcal{X}_L$ of cardinality at most v, and it outputs a bit $b \in \{0,1\}$ such that

$$\mathsf{EE}.\textsc{Gu}_k(T, Y_L, Y_R, \boldsymbol{V}) = \begin{cases} 1 & \text{if } X_L \in \boldsymbol{V}, \text{where } (X_L, X_R) \leftarrow \mathsf{EE}.\textsc{De}_k(T, Y_L, Y_R), \\ 0 & \text{otherwise.} \end{cases}$$

We assume that the distinguisher is non-trivial, meaning that for every guess query $(T, Y_L, Y_R, \boldsymbol{V})$ it makes, the triple (T, Y_L, Y_R) has not been output by the encipher oracle in some previous query, and it has not been previously queried to the decipher oracle.

In the real world, the distinguisher $\mathcal{A}$ interacts with a triplet of oracles $(\mathsf{EE}.\textsc{En}_k, \mathsf{EE}.\textsc{De}_k, \mathsf{EE}.\textsc{Gu}_k)$, with k being a uniformly sampled key. In the ideal world, the distinguisher interacts with a triplet of oracles $(\mathsf{PP}, \mathsf{PP}^{-1}, \perp)$, where $\mathsf{PP} \leftarrow\!\$\, \mathsf{TP}(\mathcal{T}, \mathcal{X}_L \times \mathcal{X}_R)$ and where $\perp$ oracle always returns 0.

Definition 2 (RPRP security). *The rugged pseudorandom permutation advantage of $\mathcal{A}$ against* EE *is defined as*

$$\mathbf{Adv}^{\mathrm{RPRP}}_{\mathsf{EE}}(\mathcal{A}, v) \triangleq \left|\Pr[\mathcal{A}^{\mathsf{EE}.\mathrm{En}_k, \mathsf{EE}.\mathrm{De}_k, \mathsf{EE}.\mathrm{Gu}_k} = 1] - \Pr[\mathcal{A}^{\mathsf{PP}, \mathsf{PP}^{-1}, \perp} = 1]\right|,$$

where the first probability is defined over the randomness of $k \leftarrow\!\!\$\, \mathcal{K}$ and the second probability is defined over the randomness of $\mathsf{PP} \leftarrow\!\!\$\, \mathsf{TP}(\mathcal{T}, \mathcal{X}_L \times \mathcal{X}_R)$.

The formal security game of the RPRP notion is shown in Figure A.1 in Appendix A.

2.3 Nonce-Based AEAD

A nonce-based *authenticated encryption* with additional data (nAEAD) scheme AE consists of a pair of deterministic algorithms $(\mathsf{Enc}, \mathsf{Dec})$ with associated key, nonce, additional data, message and ciphertext spaces $\mathcal{K}, \mathcal{N}, \mathcal{AD}, \mathcal{M}$ and $\mathcal{C}$, respectively. The encryption algorithm Enc takes in as input (K, N, A, M) and outputs a tagged ciphertext $C \in \mathcal{C}$. Similarly, the decryption algorithm takes in as input (K, N, A, C) and outputs either a plaintext $M \in \mathcal{M}$ or an error symbol $\perp$, indicating a failed decryption. We denote $\mathsf{Enc}(K, N, A, M)$ also as $\mathsf{Enc}_K(N, A, M)$, and use the analogous notation for the decryption algorithm Dec. An AEAD scheme must be *correct*, meaning that for all tuples (K, N, A, M), $\mathsf{Dec}_K(N, A, \mathsf{Enc}_K(N, A, M)) = M$.

The first AEAD security notion that we will consider in this work is the *misuse resistant authenticated encryption* (MRAE) notion. In this security notion, the adversary $\mathcal{A}$ interacts either with a pair of oracles $(\mathsf{AE}.\mathsf{Enc}_K, \mathsf{AE}.\mathsf{Dec}_K)$ in the real world, or with a pair of oracles $(\$, \perp)$ in the ideal world, where the oracle $\$$ returns random bits of length $|\mathsf{Enc}_K(N, A, M)|$ and oracle $\perp$ always returns the error symbol $\perp$.

Definition 3 (MRAE security). *The* MRAE *advantage of $\mathcal{A}$ against* AE *is defined as*

$$\mathbf{Adv}^{\mathrm{MRAE}}_{\mathsf{AE}}(\mathcal{A}) \triangleq \left|\Pr[\mathcal{A}^{\mathsf{AE}.\mathsf{Enc}_K, \mathsf{AE}.\mathsf{Dec}_K} = 1] - \Pr[\mathcal{A}^{\$, \perp} = 1]\right|,$$

where the first probability is defined over the randomness of $K \leftarrow\!\!\$\, \mathcal{K}$, and the second probability is defined over the randomness of $\$$. The adversary $\mathcal{A}$ is not allowed to forward encryption oracle responses to the decryption oracle, nor it is allowed to repeat queries to the encryption oracle.

The next security notion that we will consider is the *release of unverified plaintext* (RUPAE) notion. We consider the RUPAE notion only for *subtle AE* (SAE) syntax, which is a generalization of AE syntax and stems from the work of Barwell, Page and Stam [7]. The subtle AE syntax integrates an additional algorithm $\mathsf{Leak} : \mathcal{K} \times \mathcal{N} \times \mathcal{AD} \times \mathcal{C} \to \{\top\} \cup \mathcal{L}$, where the $\mathcal{L}$ represent the leakage set. The connection between the conventional AE and SAE syntax is, that when Leak algorithm outputs $\top$, Dec algorithm outputs a plaintext from $\mathcal{M}$. Vice versa, when Leak algorithm outputs some invalid, leaked, plaintext M, the decryption oracle outputs $\perp$.

Definition 4 (RUPAE security). *The* RUPAE *advantage of $\mathcal{A}$ against a SAE scheme* AE *is defined as*

$$\mathbf{Adv}^{\mathrm{RUPAE}}_{\mathsf{AE}}(\mathcal{A}) \triangleq \left|\Pr[\mathcal{A}^{\mathsf{AE}.\mathsf{Enc}_K, \mathsf{AE}.\mathsf{Dec}_K, \mathsf{AE}.\mathsf{Leak}_K} = 1] - \Pr[\mathcal{A}^{\$, \perp, \mathsf{S}_{leak}} = 1]\right|,$$

where the first probability is defined over the randomness of $K \leftarrow\!\!\$\, \mathcal{K}$, the S_{leak} is a stateless leakage simulator and the rest of the oracles are defined as in the AE security definition.

For additional technical details about the SAE syntax we refer the reader to [7].

2.4 H-Coefficient Technique [13, 37]

Let $\mathcal{A}$ be a computationally unbounded deterministic distinguisher that interacts with either the real oracle, i.e., the construction of our interest, or the ideal oracle which is usually considered to be a uniform random function or permutation. The collection of all the queries and responses that $\mathcal{A}$ made and received to and from the oracle, is called the *transcript* of $\mathcal{A}$, denoted as τ. Sometimes, we allow the oracle to release more internal information to $\mathcal{A}$ only after $\mathcal{A}$ completes all its queries and responses, but before it outputs its decision bit. In this case, the transcript of $\mathcal{A}$ includes the additional information about the oracle and clearly the maximum distinguishing advantage of $\mathcal{A}$ in this setting can not be less than that of

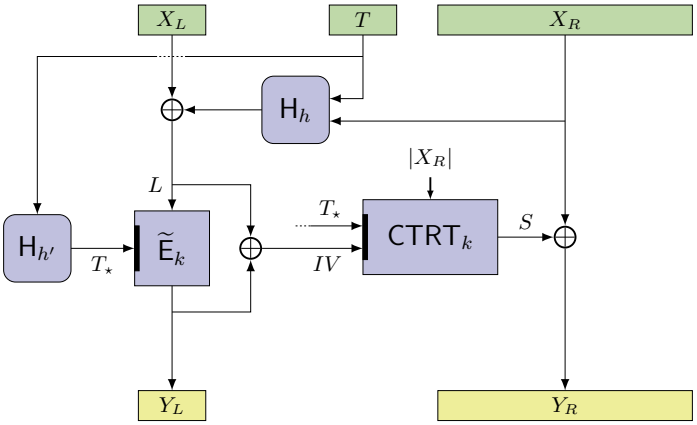


Figure 3.1: Graphical representation of the THEC enciphering algorithm.

$\mathsf{THEC}.\mathrm{En}_{k,h,h'}(T, X_L, X_R)$	$\mathsf{THEC}.\mathrm{De}_{k,h,h'}(T, Y_L, Y_R)$	Proc. $\mathsf{CTRT}_k(T_\star, IV, m)$
$T_\star \leftarrow \mathsf{H}_{h'}(T)$	$T_\star \leftarrow \mathsf{H}_{h'}(T)$	$\kappa \leftarrow \lceil m/n \rceil$
$L \leftarrow X_L \oplus \mathsf{H}_h(T, X_R)$	$L \leftarrow \widetilde{\mathsf{E}}_k^{-1}(T_\star, Y_L)$	**for** ctr $= 0$ to $\kappa - 1$
$Y_L \leftarrow \widetilde{\mathsf{E}}_k(T_\star, L)$	$IV \leftarrow L \oplus Y_L$	$S_{\mathrm{ctr}} \leftarrow \widetilde{\mathsf{E}}_k(IV \oplus \langle \mathrm{ctr} \rangle_2, T_\star)$
$IV \leftarrow L \oplus Y_L$	$S \leftarrow \mathsf{CTRT}_k(T_\star, IV, \lvert Y_R \rvert)$	$S \leftarrow (S_0, \ldots, S_{\kappa-1})$
$S \leftarrow \mathsf{CTRT}_k(T_\star, IV, \lvert X_R \rvert)$	$X_R \leftarrow Y_R \oplus S$	**return** $\lfloor S \rfloor_m$
$Y_R \leftarrow X_R \oplus S$	$X_L \leftarrow L \oplus \mathsf{H}_h(T, X_R)$	
return (Y_L, Y_R)	**return** (X_L, X_R)	

Figure 3.2: Pseudocode description of the THEC construction.

without additional information. Observe that the transcript τ is a random variable and the randomness of the distribution of τ only comes from the randomness of the oracle with which $\mathcal{A}$ interacts.

Let X_{re} (resp. X_{id}) denote the random variable that takes a transcript τ realized in the real world (resp. ideal world) while interacting with the distinguisher. A transcript τ is said to be attainable with respect to a distinguisher $\mathcal{D}$, if the probability of realizing it in the ideal world is non-zero. Let Θ denote the set of all attainable transcripts. Following these notations, we state the main theorem of the H-Coefficient technique as follows:

Theorem 1. *[**H-Coefficient**] Let $\mathcal{A}$ be a fixed deterministic distinguisher that has access to either the real oracle $\mathcal{O}_{\mathrm{re}}$ or the ideal oracle $\mathcal{O}_{\mathrm{id}}$. Let $\Theta = \mathsf{GoodT} \sqcup \mathsf{BadT}$ be some partition of the set of attainable transcripts Θ. For any good attainable transcript $\tau \in \mathsf{GoodT}$, let*

$$\frac{\mathsf{p}_{\mathrm{re}}(\tau)}{\mathsf{p}_{\mathrm{id}}(\tau)} \triangleq \frac{\Pr[\mathsf{X}_{\mathrm{re}} = \tau]}{\Pr[\mathsf{X}_{\mathrm{id}} = \tau]} \geq 1 - \epsilon_{\mathrm{good}},$$

for some $\epsilon_{\mathrm{good}} \geq 0$, and there exists $\epsilon_{\mathrm{bad}} \geq 0$ such that $\Pr[\mathsf{X}_{\mathrm{id}} \in \mathsf{BadT}] \leq \epsilon_{\mathrm{bad}}$ holds. Then,

$$\mathbf{Adv}^{\mathcal{O}_{\mathrm{id}}}_{\mathcal{O}_{\mathrm{re}}}(\mathcal{A}) \triangleq |\Pr[\mathcal{A}^{\mathcal{O}_{\mathrm{re}}} = 1] - \Pr[\mathcal{A}^{\mathcal{O}_{\mathrm{id}}} = 1]| \leq \epsilon_{\mathrm{good}} + \epsilon_{\mathrm{bad}}. \tag{1}$$

3 THEC: A BBB Secure RPRP

We present the THEC (**T**weakable **H**ash-**E**ncipher-**C**TR) construction, a tweakable block cipher based BBB-secure RPRP cipher. The graphical illustration of the enciphering algorithm of THEC is given in Figure 3.1, while its pseudocode description is given in Figure 3.2.

Our construction can be seen as a heavier variant of the HEC cipher [21], and is inspired by the THCTR construction of Dutta and Nandi [24]. However, there are three essential differences between THCTR and our THEC construction. First, THEC does not have the third, bottom, layer consisting of another hash function pass over the data. Second, we employ a slightly different counter mode in the second layer,

the reason for that being that the CTR mode used in the THCTR construction does not actually give beyond birthday bound security, as pointed out by [2,32]. The CTR mode in the THCTR construction is built upon a tweakable block cipher that takes fixed tweak for a single query and the input value to the tweakable block cipher is incremented for processing each message block. Thus, it is possible to create a tweak-output collision with a single query containing 2^{64} blocks. To circumvent this attack, we instead use[7] the GCTR-3 construction, one of the generalized counter modes proposed by Andreeva *et al.* [2]. The GCTR-3 is the generalization of the Counter-in-Tweak (CTRT) mode, introduced by Peyrin and Seurin [38], where the tweakable block cipher in the counter mode is retweaked for every input block. Andreeva *et al.* have shown that retweaking every TBC call in tweakable block cipher based counter mode is unavoidable if one wants to achieve beyond birthday bound security out of the CTR mode. The third difference in comparison to THCTR is that we do not require two different hash functions as is the case with THCTR. Moreover, the second hash function in THCTR that we were able to "get rid of" has $2n$-bit output, meaning that we were able to reduce the size of *hash keys* from $3n$ bits to $2n$ bits in total.

3.1 Security of THEC

In the following, we state the security result of THEC, proof of which is given in the following Section 4. In the following theorem and the corresponding proof, we will assume the adversary only makes inputs that "end on a full block", i.e. for an input (T, X_L, X_R) it holds $|X_R| \pmod n = 0$. However, the proof can be extended to admit inputs of any bit-length (cf. [21]).

Theorem 2. *Let* THEC *be the construction defined in Figure 3.2 over the split domain* $\{0,1\}^n \times \{0,1\}^*$ *and assume that* H *is* ϵ*-AXU,* ρ*-almost regular and* δ*-AU* [8]*. Then, given any adversary* $\mathcal{A}$ *making* q_{en} *encipher,* q_{de} *decipher and* q_{gu} *guess queries, with each query being at most* l_{m} *blocks long and* $\mathcal{A}$ *querying* σ *blocks in total, there exists an adversary* $\mathcal{B}$ *such that, under the assumption that* $q := (q_{\text{en}} + q_{\text{de}} + q_{\text{gu}}) \leq 2^{n-1}$, $\mu \leq q$ *and* $q_{\text{gu}} v \leq 2^{n-1}$*, where* μ *denotes the maximum number of tweak repetitions, it holds that*

$$\begin{aligned}\mathbf{Adv}^{\text{RPRP}}_{\mathsf{THEC}}(\mathcal{A}, v) \leq\ & \mathbf{Adv}^{\text{STPRP}}_{\widetilde{\mathsf{E}}}(\mathcal{B}) + q(\mu-1)\epsilon + q^2\delta\epsilon + \frac{q(\mu-1)}{2^{n-1}} + \frac{q^2\delta}{2^{n-1}} \\ & + \mu\sigma\rho + q\sigma\rho^2 + \frac{q\sigma\rho}{2^{n-1}} + 2\sigma\rho + \frac{q(\mu-1)l_{\text{m}}}{2^{n-1}} + \frac{\min\{q^2 l_{\text{m}}, \sigma^2\}\delta}{2^{n-1}} \\ & + \frac{\min\{q^2 l_{\text{m}}, \sigma^2\}}{2^{2n-2}} + q_{\text{gu}} v\left(\epsilon + \frac{2}{2^{n-1}}\right) + (\mu-1)\frac{q}{2^n}.\end{aligned}$$

4 Proof of Theorem 2 (THEC Construction)

We replace the TBC $\widetilde{\mathsf{E}}$ in THEC with an n-bit tweakable random permutation $\widetilde{\mathsf{P}}$ at the cost of STPRP security of $\widetilde{\mathsf{E}}$ and call the resulting construction THEC*. We employ the H-Coefficient technique to bound the RPRP distinguishing advantage of THEC*. The real world is THEC* and the ideal world consists of the pair of oracles $(\mathsf{PP}, \mathsf{PP}^{-1})$, where $\mathsf{PP} \leftarrow\!\!\$\ \mathsf{TP}(\mathcal{T}, \mathcal{X}_L \times \mathcal{X}_R)$. The adversary summarizes the query-response pair in transcript τ', where τ' contains all En, De and Gu queries the adversary made during the interaction.

4.1 Extended Query Transcript

After the interaction (before outputting the decision bit), the adversary is provided with the hash keys h, h' used in the construction. However, in the ideal world, they are sampled uniformly and independently from the hash key space $\mathcal{H} \times \mathcal{H}$. Once the hash keys are revealed, the adversary can compute $T^i_* \leftarrow \mathsf{H}_{h'}(T^i)$ for all $i \in [q_{\text{en}} + q_{\text{de}} + q_{\text{gu}}]$. The adversary is also provided with the extended guess oracle query-response pairs that release the values (X_L, X_R). Therefore, an i-th guess oracle query transcript will be of the form $(T^i, Y^i_L, Y^i_R, \boldsymbol{V}^i, \mathsf{r}^i, X^i_L, X^i_R)$, where $\boldsymbol{V}^i$ and r^i represent the set of guessed values and the response of the guess oracle, respectively. Now, we define the following:

[7] For this, and our other construction as well.

[8] ϵ-AXU is the maximum probability of the hash difference of two distinct messages attain a particular value. Similarly, δ-AU is the maximum probability of two hash values colliding for two distinct messages and finally ρ-almost regular is the maximum probability that the hash output attains a particular value.

Definition 5. *A guess oracle query $(T^i, Y_L^i, Y_R^i, \boldsymbol{V}^i, \mathsf{r}^i, X_L^i, X_R^i)$ is fresh, if its triple (T^i, Y_L^i, Y_R^i) has been queried for the first time. A fresh guess oracle query $(T^i, Y_L^i, Y_R^i, \boldsymbol{V}^i, \mathsf{r}^i, X_L^i, X_R^i)$ is called independent, if there is no j-th encipher or decipher query, with $i \neq j$, or j-th guess query, with $j < i$, such that*

$$T_\star^i = T_\star^j \quad \textit{and} \quad Y_L^i = Y_L^j.$$

A fresh guess oracle query that is not independent is called a dependent guess oracle query.

In the ideal world, we sample (X_L^i, X_R^i) as follows:

1. i-th independent Gu query: Sample $(X_L^i, X_R^i) \leftarrow\!\$\, \mathsf{PP}^{-1}(T^i, \cdot, \cdot)$, and then set $\mathsf{PP}(T^i, X_L^i, X_R^i) \leftarrow (Y_L^i, Y_R^i)$.
2. i-th dependent Gu query: We find the longest query (say j-th) that the i-th query depends on. Then set
$$X_R^i \leftarrow \begin{cases} Y_R^i \oplus \lfloor Y_R^j \oplus X_R^j \rfloor_{|Y_R^i|}, \text{ if } \left|Y_R^i\right| \leq \left|Y_R^j\right| \\ Y_R^i \oplus \left((Y_R^j \oplus X_R^j) \| U^i\right) \text{ where } U^i \leftarrow\!\$\, \{0,1\}^{|Y_R^i| - |Y_R^j|}, \text{ otherwise} \end{cases}$$
$$X_L^i \leftarrow \mathsf{H}_h(T^i, X_R^i) \oplus \underbrace{X_L^j \oplus \mathsf{H}_h(T^j, X_R^j)}_{L^j}.$$
3. i-th non-fresh Gu query: Set (X_L^i, X_R^i) to (X_L^j, X_R^j) of a previous guess oracle query (i.e., $j < i$) that had the same triple (T^i, Y_L^i, Y_R^i) queried.

4.2 Defining and Bounding Bad Transcripts

Let Θ be the set of all attainable transcripts. We say an attainable transcript is **bad** if (a) it is either involved in non-trivial collisions in the tweak-input or tweak-output pairs for any call to the tweakable block cipher of the construction or (b) it is involved in a successful guess oracle query. In the following definition, κ denotes the number of n-bit blocks in the right part of the query, i.e. X_R or Y_R, and $q_{\text{gu-ind}}$ and $q_{\text{gu-dep}}$ represent the number of independent and dependent guess oracle queries, respectively.

Definition 6. *An attainable transcript $\tau = (\tau', h, h')$ is bad, if there exist two queries i and j, such that either of the following holds:*

[B1] $\exists i, j \in [q_{\text{en}} + q_{\text{de}} + q_{\text{gu-ind}}], i \neq j$: $T_\star^i = T_\star^j, X_L^i \oplus \mathsf{H}_h(T^i, X_R^i) = X_L^j \oplus \mathsf{H}_h(T^j, X_R^j)$.

[B2] $\exists i, j \in [q_{\text{en}} + q_{\text{de}} + q_{\text{gu-ind}}], i \neq j$: $T_\star^i = T_\star^j, Y_L^i = Y_L^j$.

[B3] $\exists i, j \in [q_{\text{en}} + q_{\text{de}} + q_{\text{gu-ind}}]$ *and* $\text{ctr}_j \in [\kappa_j]$: $T_\star^i = IV^j \oplus \text{ctr}_j, X_L^i \oplus \mathsf{H}_h(T^i, X_R^i) = T_\star^j$.

[B4] $\exists i, j \in [q_{\text{en}} + q_{\text{de}} + q_{\text{gu-ind}}]$ *and* $\text{ctr}_j \in [\kappa_j]$: $T_\star^i = IV^j \oplus \text{ctr}_j, Y_L^i = X_R^j[\text{ctr}_j] \oplus Y_R^j[\text{ctr}_j]$.

[B5] *Either* $(\exists i, j \in [q_{\text{en}} + q_{\text{de}} + q_{\text{gu-ind}}], i \neq j)$ *or* $(\exists i \in [q], j \in [q_{\text{gu-dep}}]$*, such that j does not depend on query i), and* $\exists \text{ctr}_i \in [\kappa_i], \text{ctr}_j \in [\kappa_j]$: $IV^i \oplus \text{ctr}_i = IV^j \oplus \text{ctr}_j$ *and* $T_\star^i = T_\star^j$.

[B6] *Either* $(\exists i, j \in [q_{\text{en}} + q_{\text{de}} + q_{\text{gu-ind}}], i \neq j)$ *or* $(\exists i \in [q], j \in [q_{\text{gu-dep}}]$*, such that j does not depend on query i), and* $\exists \text{ctr}_i \in [\kappa_i], \text{ctr}_j \in [\kappa_j]$: $IV^i \oplus \text{ctr}_i = IV^j \oplus \text{ctr}_j, X_R^i[\text{ctr}_i] \oplus Y_R^i[\text{ctr}_i] = X_R^j[\text{ctr}_j] \oplus Y_R^j[\text{ctr}_j]$,

[B7] $\exists i \in [q_{\text{gu}}]$: $X_L^i \in \boldsymbol{V}^i$ *(i.e., a successful guess oracle query)*

Lemma 1. *Let* BadT *be the set of bad (defined as above) attainable transcripts. Then, it holds*

$$\Pr[\mathsf{X}_{\text{id}} \in \mathsf{BadT}] \leq \epsilon_{\text{bad}} \leq q(\mu-1)\epsilon + q^2\delta\epsilon + \frac{q(\mu-1)}{2^{n-1}} + \frac{q^2\delta}{2^{n-1}} + \mu\sigma\rho + q\sigma\rho^2 + \frac{q\sigma\rho}{2^{n-1}} + 2\sigma\rho + \frac{q(\mu-1)l_{\text{m}}}{2^{n-1}} + \frac{\min\{q^2 l_{\text{m}}, \sigma^2\}\delta}{2^{n-1}} + \frac{\min\{q^2 l_{\text{m}}, \sigma^2\}}{2^{2n-2}} + q_{\text{gu}} v \left(\epsilon + \frac{2}{2^{n-1}}\right).$$

To prove Lemma 1 and bound the probabilities that the bad events happen, let τ be some attainable transcript.

Bounding [B1]: If $T^i = T^j$, the calculation reduces to bounding the probability that $\mathsf{H}_h(T^i, X_R^i) \oplus \mathsf{H}_h(T^j, X_R^j) = X_L^j \oplus X_L^i$, which holds true with probability at most ϵ. Summing over all $i \neq j$, the bound becomes $q(\mu-1)\epsilon$, where recall that μ denotes the maximum number of tweak repetitions and ϵ is the AXU advantage of the hash function H. If $T^i \neq T^j$, the probability that [B1] holds true is bounded by $\delta\epsilon$, since we can bound the first and second equation with AU and AXU advantage of H, respectively. Therefore, summing over all $i \neq j$, the bound becomes at most $q^2\delta\epsilon$.

In total, the probability that condition [B1] holds true is upper bounded by

$$\Pr[\text{B1}] \leq q(\mu-1)\epsilon + q^2\delta\epsilon. \tag{2}$$

Bounding [B2]: Following the definition of independent guess queries, we can disregard the case where i-th or j-th query is a guess oracle query. This follows because an independent guess oracle query will never fulfill the bad condition [B2] in combination with any other query. If $T^i = T^j$, bounding the probability of the event [B2] reduces to bounding the probability of the event $Y_L^i = Y_L^j$. Without loss of generality, we assume that $i < j$. Following the definition of RPRP, (a) both i-th query and j-th query cannot be decipher queries and (b) j-th query cannot be the decipher one. Therefore, j-th query must be an encipher one. Assuming $q \leq 2^{n-1}$, we upper bound the probability that $Y_L^i = Y_L^j$ holds true with 2^{n-1}. The calculation, which is used in many places in this work, is as follows.

$$\frac{2^{l_j-n}}{2^{l_j}-(j-1)} \leq \frac{2^{l_j-n}}{2^{l_j}-q} \leq \frac{2^{l_j-n}}{2^{l_j-1}} = \frac{1}{2^{n-1}},$$

where the denominator corresponds to the minimal size of the sample space (i.e., non-defined values of the permutation $\widetilde{\mathsf{P}}(T^i,\cdot,\cdot)$), while the numerator is the maximal sample size. Note that $i < j$, and the response (Y_L^j, Y_R^j) to the j-th query was sampled after i-th oracle query was already fixed. Summing over all $i \neq j$, the bound when $T^i = T^j$ becomes $q(\mu-1)/2^{n-1}$.
If $T^i \neq T^j$, then we bound the first equation by δ and second equation again by $\frac{1}{2^{n-1}}$. Summing over $i \neq j$, the bound becomes $\frac{q^2\delta}{2^{n-1}}$. Therefore, by combining the above two bounds, the probability that the event [B2] holds is upper bounded by

$$\Pr[\text{B2}] \leq \frac{q(\mu-1)}{2^{n-1}} + \frac{q^2\delta}{2^{n-1}}. \tag{3}$$

Bounding [B3]: Fix some i, j and ctr_j. We upper bound the probability of the event in the following two cases:

Case I: $i \neq j$**.** The first equation, $T_\star^i = IV^j \oplus \mathrm{ctr}_j$, holds with probability at most ρ, where ρ is the almost-regular advantage of H. If $T^i = T^j$, the second equation is bounded with probability 1 and summing up over all $i \neq j$ and ctr_j, where $T^i = T^j$, we get an upper bound of $\sigma(\mu-1)\rho$. Otherwise $T^i \neq T^j$, and in this subcase we can bound the second equation with ρ as well. In total, over all $i \neq j$ and ctr_j, where $T^i \neq T^j$, we get an upper bound of $q\sigma\rho^2$.
Case II: $i = j$**.** The first equation, $T_\star^i = IV^j \oplus \mathrm{ctr}_j$, holds with probability at most ρ, where ρ is the almost-regular advantage of H. We can bound the second equation with 1, and summing up over all $i = j$ and ctr_j, the probability of the event [B3] holds true is at most $\sigma\rho$.

By combining the above two cases, probability that the event [B3] holds true is at most

$$\Pr[\text{B3}] \leq \sigma(\mu-1)\rho + q\sigma\rho^2 + \sigma\rho = \mu\sigma\rho + q\sigma\rho^2. \tag{4}$$

Bounding [B4]: Fix some i, j and ctr_j. We upper bound the probability of the event in the following two cases.

Case I: $i \neq j$**.** The first equation, when expanded, is equivalent to

$$T_\star^i = X_L^j \oplus \mathsf{H}_h(T^j, X_R^j) \oplus Y_L^j \oplus \mathrm{ctr}_j,$$

and the probability that the above equation holds true is bounded by the almost-regular advantage ρ of the underlying hash function H. On the other hand, to bound the second equation, $Y_L^i = X_R^j[\mathrm{ctr}_j] \oplus Y_R^j[\mathrm{ctr}_j]$, we need to apply a more careful argument.
If $i < j$, we can bound the equation by randomness of $Y_R^j[\mathrm{ctr}_j]$ (in case j-th query is an encipher query), or by randomness of $X_R^j[\mathrm{ctr}_j]$ (in case it is a decipher or an independent guess oracle query).
If $i > j$, we differentiate two subcases: (a) query i is an encipher query and (b) query i is a decipher or independent guess oracle query. In subcase (a), the second equation can be upper bounded by $\frac{1}{2^{n-1}}$, due to randomness of Y_L^i. In subcase (b), there is no way to bound the second equation since, e.g., the adversary can always make a decipher query with Y_L^i such that the equation holds true. However, we claim that, in subcase (b), for every choice of j and ctr_j, there is only single such query

i that would be a valid query. Indeed, if i is a decipher query, Y_L^i can be only repeated once. If i is an independent guess oracle query, there can be no other independent guess query i' that also satisfies the first and second equation since then either i or i' would not be an independent guess oracle query (i.e., we have a contradiction). Summarizing the calculations in case I, we have that upper bound for event [B4] holding true is

$$\begin{aligned} \frac{q\sigma\rho}{2^{n-1}}, &\quad \text{if } i < j \text{ or } i > j \text{ (subcase (a)), or} \\ \sigma\rho, &\quad \text{if } i > j \text{ (subcase (b))} \end{aligned}$$

Case II: $i = j$**.** The first equation can be bounded, as in the case I, with ρ. The second equation in this case we can bound with 1. Then, summing up over all i and ctr_i, the bound in this case is $\sigma\rho$.

Therefore, by combining the bounds from case I and II, the probability that the event [B4] holds true is upper bounded by

$$\Pr[\text{B4}] \leq \frac{q\sigma\rho}{2^{n-1}} + 2\sigma\rho. \tag{5}$$

Bounding [B5]: There are two possibilities for choices of i and j in the event [B5], either (i) $\exists i, j \in [q_{\text{en}} + q_{\text{de}} + q_{\text{gu-ind}}], i \neq j$, or (ii) $\exists i \in [q], j \in [q_{\text{gu-dep}}]$, such that j does not depend on query i. The probability in both instances is calculated analogously, thus in the following we give explicitly the calculation for the instance (i), but sum over i, j taking both instances into account.

Fix some $i \neq j$, ctr_i and ctr_j. Without loss of generality, we assume that $i < j$.

Case I: $T^i = T^j$**.** In this case, bounding the probability of the event that [B5] holds true reduces to bounding the probability that $IV^i \oplus \mathrm{ctr}_i = IV^j \oplus \mathrm{ctr}_j$ holds true. By expanding the equation and shuffling the summands, we get

$$X_L^i \oplus \mathsf{H}_h(X_R^i) \oplus Y_L^i \oplus X_L^j \oplus \mathsf{H}_h(X_R^j) \oplus Y_L^j = \mathrm{ctr}_i \oplus \mathrm{ctr}_j.$$

This equation can be bounded by the randomness of X_L^j, if j-th query was a decipher or independent guess query, or by the randomness of Y_L^j, if j-th query was an encipher query. Thus, the probability that the equation above holds is $\frac{1}{2^{n-1}}$. Summing up over all $i \neq j$ and $\mathrm{ctr}_i \oplus \mathrm{ctr}_j$ the upper bound in this case is $\frac{q(\mu-1)l_{\text{m}}}{2^{n-1}}$.

Case II: $T^i \neq T^j$**.** In this case, bounding the probability of the event that [B5] holds true, reduces to bounding the probability of the event that the first equation $IV^i \oplus \mathrm{ctr}_i = IV^j \oplus \mathrm{ctr}_j$ holds true and bounding the probability for the second equation, $T_\star^i = T_\star^j$ holds true. Bounding the probability of the first equation holding true is identical to the case I above. However, the probability of the second equation holding true is bounded above by AU advantage δ of the underlying hash function H. For fixed indices i and j, the number of choices for $\mathrm{ctr}_i \oplus \mathrm{ctr}_j$ is l_{m}. Alternatively, we can also count over the number of (i, ctr_i) and (j, ctr_j) pairs. Thus, by summing up over all i, j, ctr_i and ctr_j, the upper bound in this case is $\frac{\min\{q^2 l_{\text{m}}, \sigma^2\}\delta}{2^{n-1}}$.

By combining the above two bounds, the total probability that the event [B5] holds true is upper bounded by

$$\Pr[\text{B5}] \leq \frac{q(\mu-1)l_{\text{m}}}{2^{n-1}} + \frac{\min\{q^2 l_{\text{m}}, \sigma^2\}\delta}{2^{n-1}}. \tag{6}$$

Bounding [B6]: Similarly to the event [B5], there are two possibilities for choices of i and j in the event [B6]. The probability in both instances is calculated analogously, thus in the following we give explicitly the calculation for the first instance, but sum over i, j taking both instances into account.

Fix some $i \neq j$, ctr_i and ctr_j. Without loss of generality, we assume that $i < j$. To bound the event, we note that the probability of the first equation, $IV^i \oplus \mathrm{ctr}_i = IV^j \oplus \mathrm{ctr}_j$, is bounded above by $\frac{1}{2^{n-1}}$. The probability of the second equation, i.e., $X_R^i[\mathrm{ctr}_i] \oplus Y_R^i[\mathrm{ctr}_i] = X_R^j[\mathrm{ctr}_j] \oplus Y_R^j[\mathrm{ctr}_j]$ is bounded above by $\frac{1}{2^{n-1}}$, due to the randomness of $Y_R^j[\mathrm{ctr}_j]$ (in case j-th query is an encipher query), or due to the randomness of $X_R^j[\mathrm{ctr}_j]$ (in case it is a decipher or an independent guess oracle query). Summing up over all i, j, ctr_i and ctr_j, the probability that the event [B6] holds true is upper bounded by

$$\Pr[\text{B6}] \leq \frac{\min\{q^2 l_{\text{m}}, \sigma^2\}}{2^{2n-2}}. \tag{7}$$

Bounding [B7]: In order to calculate the probability of a successful guess query, we split all guess oracle queries into equivalence classes, where two guess oracle queries $(T^i, Y_L^i, Y_R^i, \boldsymbol{V}^i, \mathsf{r}^i, X_L^i, X_R^i)$ and $(T^j, Y_L^j, Y_R^j, \boldsymbol{V}^j, \mathsf{r}^j, X_L^j, X_R^j)$ are in the same equivalence class if and only if $(T^i, Y_L^i, Y_R^i) = (T^j, Y_L^j, Y_R^j)$. It follows that an equivalence class will contain all guess queries with some fixed triple (T, Y_L, Y_R). Let q'_{gu} be the number of such equivalence classes and let q^c_{gu} denote the number of queries in c-th equivalence class. Note that, the first element in every equivalence class is the first time the guess oracle was queried with the corresponding triple (T, Y_L, Y_R). Now, we have the following two cases: (I) the first query is an independent guess oracle query or (II) it is a dependent guess oracle query.

Case I: Independent guess oracle query. This means the pair (X_L, X_R) corresponding to the triple (T, Y_L, Y_R) has been sampled according to the ideal world distribution. To calculate the probability of a successful guess query, we collect all guesses the adversary made for queries in the c-th equivalence class. Then, we use the principle of deferred decision to calculate the probability that sampled X_L matches one of the guesses by "delaying" the sampling of (X_L, X_R) to the time the last guess query in this equivalence class was made. We can do that since, although the adversary's choices of guess sets $\boldsymbol{V}$ can be influenced during the game by other queries the adversary is making, the sampling of (X_L, X_R) can be deferred to a later point with a side-effect of sample space being smaller. To conclude, for c-th equivalence class, the probability that $\mathcal{A}$ made a successful guess query, assuming $q \leq 2^{n-1}$, can be bounded by

$$\frac{q^c_{\mathrm{gu}} v\, 2^{l_i - n}}{2^{l_i} - (i-1)} \leq \frac{q^c_{\mathrm{gu}} v\, 2^{l_i - n}}{2^{l_i} - q} \leq \frac{q^c_{\mathrm{gu}} v\, 2^{l_i - n}}{2^{l_i - 1}} = \frac{q^c_{\mathrm{gu}} v}{2^{n-1}},$$

where i-th query is the last query in equivalence class c, and l_i denotes the length of the queries in c-th equivalence class.

Case II: Dependent guess oracle query. This means the pair (X_L, X_R) corresponding to the triple (T, Y_L, Y_R) was set by the simulator, more concretely the left value X_L gets the value of $\mathsf{H}_h(T^i, X_R^i) \oplus X_L^j \oplus \mathsf{H}_h(T^j, X_R^j)$, where j-th query is the query the first query in the equivalence class depends on. We note the value $\mathsf{H}_h(T^i, X_R^i)$ is the same for all queries in this equivalence class. To calculate probability of a successful guess query in this subcase, we fix some guess value X_L^* from the guess set in a first guess oracle query in the respective equivalence class. We then calculate the probability that

$$X_L^* = \mathsf{H}_h(T^i, X_R^i) \oplus X_L^j \oplus \mathsf{H}_h(T^j, X_R^j), \tag{8}$$

and then sum over all guess values in the first query and in the rest of the queries of the equivalence class. Let us call the event that the equation above holds CORRG. In addition, we define several more events: DEP_1 is an event that j-th query is a *previous* encipher, decipher or guess oracle query, DEP_2 is an event that j-th query is a *later* encipher query and DEP_3 is an event that j-th query is a *later* decipher query. Utilizing the law of total probability, we have that

$$\begin{aligned}\Pr[\mathrm{CORRG}] &= \sum_{i=1}^{3} \Pr[\mathrm{CORRG} \mid \mathrm{DEP}_i] \Pr[\mathrm{DEP}_i] \\ &\leq \sum_{i \in \{1,3\}} \Pr[\mathrm{CORRG} \mid \mathrm{DEP}_i] + \Pr[\mathrm{DEP}_2].\end{aligned}$$

Now we calculate the bound of each term independently.

- $\Pr[\mathrm{CORRG} \mid \mathrm{DEP}_1]$. If $(T^i, X_R^i) \neq (T^j, X_R^j)$ we can bound the term by ϵ. Otherwise the two pairs are equal and the Eqn. (8) reduces to $X_L^* = X_L^j$. The probability that $X_L^* = X_L^j$, given that $T^i = T^j, Y_L^i = Y_L^j$ and $Y_R^i = Y_R^j$, is 0. Since in this case j-th query is an encipher query, such a guess oracle query would be forbidden by the game. Therefore, we bound $\Pr[\mathrm{CORRG} \mid \mathrm{DEP}_1]$ with ϵ.
- $\Pr[\mathrm{DEP}_2]$. This is equal to the probability that $T_\star^i = T_\star^j$ and $Y_L^i = Y_L^j$. We bound this simply with $\Pr\left[Y_L^i = Y_L^j\right]$. Since Y_L^j was sampled after Y_L^i, we know it holds $\Pr\left[Y_L^i = Y_L^j\right] \leq \frac{1}{2^{n-1}}$.
- $\Pr[\mathrm{CORRG} \mid \mathrm{DEP}_3]$. Since j-th query is a decipher query, we know X_L^j was sampled the last, and the probability that equality holds can be bounded with $\frac{1}{2^{n-1}}$.

Now we can sum the $\Pr[\mathrm{CORRG}]$ over *all* guesses and queries in c-th equivalence class and bound the probability of a successful guess query in c-th equivalence class in this subcase with

$$q^c_{\mathrm{gu}} v \left(\epsilon + \frac{2}{2^{n-1}}\right).$$

Finally, the probability that bad condition [B7] occurs can be calculated as follows

$$\begin{aligned}\Pr[\text{B7}] &\leq \sum_{i=1}^{q'_{\text{gu}}} q_{\text{gu}}^i v \left(\epsilon + \frac{2}{2^{n-1}}\right) = v\left(\epsilon + \frac{2}{2^{n-1}}\right) \sum_{i=1}^{q'_{\text{gu}}} q_{\text{gu}}^i \\ &= q_{\text{gu}} v \left(\epsilon + \frac{2}{2^{n-1}}\right).\end{aligned} \tag{9}$$

Summing up the bounds in Eqn. (2)-(7) and Eqn. (9), we get

$$\begin{aligned}\epsilon_{\text{bad}} \leq\ & q(\mu-1)\epsilon + q^2\delta\epsilon + \frac{q(\mu-1)}{2^{n-1}} + \frac{q^2\delta}{2^{n-1}} + \mu\sigma\rho + q\sigma\rho^2 + \frac{q\sigma\rho}{2^{n-1}} \\ &+ 2\sigma\rho + \frac{q(\mu-1)l_{\text{m}}}{2^{n-1}} + \frac{\min\{q^2 l_{\text{m}}, \sigma^2\}\delta}{2^{n-1}} + \frac{\min\{q^2 l_{\text{m}}, \sigma^2\}}{2^{2n-2}} + q_{\text{gu}} v \left(\epsilon + \frac{2}{2^{n-1}}\right).\end{aligned}$$

4.3 Analysis of Good Transcripts

Now we lower bound the ratio of the real to ideal interpolation probability for a good transcript. We being by introducing a few notations needed for calculating the ratio.

- σ' denotes the total number of calls to the tweakable permutation $\widetilde{\mathsf{P}}$ in En, De, independent Gu oracle queries and calls to $\widetilde{\mathsf{P}}$ that generate the "independent"/new part of the key stream in *dependent* guess queries that are longer than the queries they depend on. Additionally, let σ_{d} be the total number of such "independent" key stream blocks for dependent oracle queries longer than the queries they depend on.
- $\mathcal{T}$ denotes the set of distinct tweaks queried to the tweakable permutation $\widetilde{\mathsf{P}}$.
- σ'_T, $T \in \mathcal{T}$, denotes the number of calls to the tweakable permutation $\widetilde{\mathsf{P}}$ with tweak T.
- L denotes the number of distinct input lengths with input lengths being $l_1, \ldots, l_L$.
- c_i denotes the number of En, De and independent Gu queries of length l_i, while C_i denotes the number of distinct tweaks queried for inputs of length l_i.
- $c_{i,j}$ denotes the number of times j-th tweak (out of C_i tweaks) has been queried (in En, De or independent Gu query) for inputs of length l_i.

Therefore, it holds that,

$$\sigma' = \sum_{T \in \mathcal{T}} \sigma'_T,\ c_i = \sum_{j=1}^{C_i} c_{i,j},\ \sigma' = \sum_{i=1}^{L} c_i \kappa_i + \sigma_{\text{d}}, \tag{10}$$

where each input length l_i consists of κ_i blocks, i.e. $l_i = \kappa_i n$. Note that the interpolation probability for dependent guess oracle queries, that have shorter or equal inputs than the query they depend on, will always be one in both worlds as the values in such dependent guess oracle queries in an attainable transcript are already determined by the query they depend on and the sampled hash keys. We take care of the rest of the dependent guess oracle queries implicitly with σ' and σ_{d} variables.

Interpolation Probability in the Ideal World. In the ideal world, (Y_L^i, Y_R^i) is uniformly sampled as a response to an encipher query. Similarly, (X_L^i, X_R^i) is uniformly sampled as a response to a decipher query or an independent guess oracle query. Finally, the challenger in the ideal world samples the hash keys uniformly at random from the hash key space $\mathcal{H} \times \mathcal{H}$. Thus, the interpolation probability in the ideal world is

$$\Pr[X_i = \tau] = \frac{1}{|\mathcal{H}|^2} \times \prod_{i=1}^{L} \prod_{j=1}^{C_i} \frac{1}{(2^{l_i})_{c_{i,j}}} \times \frac{1}{2^{n\sigma_{\text{d}}}}. \tag{11}$$

Interpolation Probability in the Real World. We calculate the interpolation probability for a good transcript τ. We know that none of the bad conditions will occur for such a good transcript. Recall that the events [B1]-[B6] exclude collisions in input or output of the tweakable permutation $\widetilde{\mathsf{P}}$ for some fixed tweak queried to it. Hence, to calculate the real world interpolation probability, we count the number of times the underlying tweakable permutation $\widetilde{\mathsf{P}}$ has been invoked. Since the hash keys have been sampled uniformly at random from the hash key space $\mathcal{H} \times \mathcal{H}$, the interpolation probability for the good transcript τ in the real world is

$$\Pr[X_r = \tau] = \frac{1}{|\mathcal{H}|^2} \times \prod_{T \in \mathcal{T}} \frac{1}{(2^n)_{\sigma'_T}}. \tag{12}$$

Ratio of Real to Ideal Interpolation Probability. By taking the ratio of Eqn. (12) to Eqn. (11), and applying the fact that $c_{i,j} \leq \mu$ for any i, j, and $(2^n)_{\sigma'_T} \leq (2^n)^{\sigma'_T}$ we have

$$\frac{\Pr[X_r = \tau]}{\Pr[X_i = \tau]} = \frac{2^{n\sigma_\mathrm{d}} \times \prod_{i=1}^{L} \prod_{j=1}^{C_i} \left(2^{l_i}\right)_{c_{i,j}}}{\prod_{T \in \mathcal{T}} (2^n)_{\sigma'_T}} \geq \frac{2^{n\sigma_\mathrm{d}} \times \prod_{i=1}^{L} \prod_{j=1}^{C_i} (2^{l_i} - (\mu-1))^{c_{i,j}}}{\prod_{T \in \mathcal{T}} (2^n)^{\sigma'_T}}$$
$$\geq \frac{2^{n\sigma_\mathrm{d}} \times \prod_{i=1}^{L} (2^{l_i} - (\mu-1))^{\sum_{j=1}^{C_i} c_{i,j}}}{\prod_{T \in \mathcal{T}} (2^n)^{\sigma'_T}}.$$

Furthermore, it holds

$$\frac{\Pr[X_r = \tau]}{\Pr[X_i = \tau]} \overset{(1)}{\geq} \frac{2^{n\sigma_\mathrm{d}} \times \prod_{i=1}^{L} (2^{l_i} - (\mu-1))^{c_i}}{2^{n\sigma'}} \overset{(2)}{\geq} \frac{2^{n\sigma_\mathrm{d}} \times \prod_{i=1}^{L} (2^{l_i} - (\mu-1))^{c_i}}{2^{n \sum_{i=1}^{L} c_i \kappa_i} \times 2^{n\sigma_\mathrm{d}}},$$

where inequalities (1) and (2) follow from Eqn. (10). Cancelling $2^{n\sigma_\mathrm{d}}$, rewriting l_i as $n\kappa_i$ and reordering the terms a bit leads us to have

$$\frac{\Pr[X_r = \tau]}{\Pr[X_i = \tau]} \geq \prod_{i=1}^{L} \left(\frac{2^{n\kappa_i} - (\mu-1)}{2^{n\kappa_i}}\right)^{c_i} = \prod_{i=1}^{L} \left(1 - \frac{(\mu-1)}{2^{n\kappa_i}}\right)^{c_i}.$$

Finally, applying Weierstrass inequality yields

$$\frac{\Pr[X_r = \tau]}{\Pr[X_i = \tau]} \geq 1 - \sum_{i=1}^{L} \frac{(\mu-1)c_i}{2^{n\kappa_i}} \geq 1 - (\mu-1)\frac{\sum_{i=1}^{L} c_i}{2^n} \geq 1 - \underbrace{(\mu-1)\frac{q}{2^n}}_{\epsilon_{\mathrm{good}}}.$$

By combining Theorem 1, Lemma 1 and ϵ_{good}, the result of Theorem 2 follows. □

5 DE-THEC: Domain Extended THEC

We present the DE-THEC (**D**omain-**E**xtended **T**weakable **H**ash-**E**ncipher-**C**TR) construction, an optimally secure RPRP cipher. We note that the DE-THEC construction is a variant of the THEC construction where the left part of the construction replaces the tweakable block cipher with a 3-round tweakable block cipher based Luby-Rackoff construction [16][9]. The graphical illustration of the enciphering algorithm of DE-THEC is given in Figure 5.1, while its pseudocode description is given in Figure 5.2.

5.1 Security of DE-THEC

In the following theorem, we state the security result of DE-THEC, proof of which is deferred to the following Section 6. We again assume, as we did for the proof of THEC's security, that the adversary only makes inputs that "end on a full block", i.e. for an input (T, X_L, X_R) it holds $|X_R| \pmod n = 0$.

Theorem 3. *Let* DE-THEC *be the construction defined in Figure 5.2 over the split domain* $\{0,1\}^{2n} \times \{0,1\}^*$ *and assume that* $\mathsf{H}' : \mathcal{H}' \times \{0,1\}^* \to \{0,1\}^{2n}$ *is an* ϵ*-AXU keyed hash function. Then, given any adversary* $\mathcal{A}$ *making* q_{en} *encipher,* q_{de} *decipher and* q_{gu} *guess queries, with each query being at most* l_{m} *blocks long and* $\mathcal{A}$ *querying* σ *blocks in total, there exists an adversary* $\mathcal{B}$ *such that, under the assumption that* $q := (q_{\mathrm{en}} + q_{\mathrm{de}} + q_{\mathrm{gu}}) \leq 2^{n-1}$ *and* $q_{\mathrm{gu}} v \leq 2^{n-1}$*, it holds that*

$$\mathbf{Adv}^{\mathrm{RPRP}}_{\text{DE-THEC}}(\mathcal{A}, v) \leq 4\mathbf{Adv}^{\mathrm{STPRP}}_{\widetilde{\mathsf{E}}}(\mathcal{B}) + q^2\epsilon + \frac{2qq_{\mathrm{en}}}{2^{2n-2}} + \frac{qq_{\mathrm{de}}}{2^{2n-2}} + \frac{2\min\{q^2 l_{\mathrm{m}}, \sigma^2\}}{2^{2n-2}} + q_{\mathrm{gu}} v \left(2\epsilon + \frac{1}{2^{2n-2}}\right) + \frac{q^2}{2^{2n}}.$$

[9] This idea of extending the left domain of a RPRP, also inspired by [16], already appeared in [21], albeit in a more generic format.

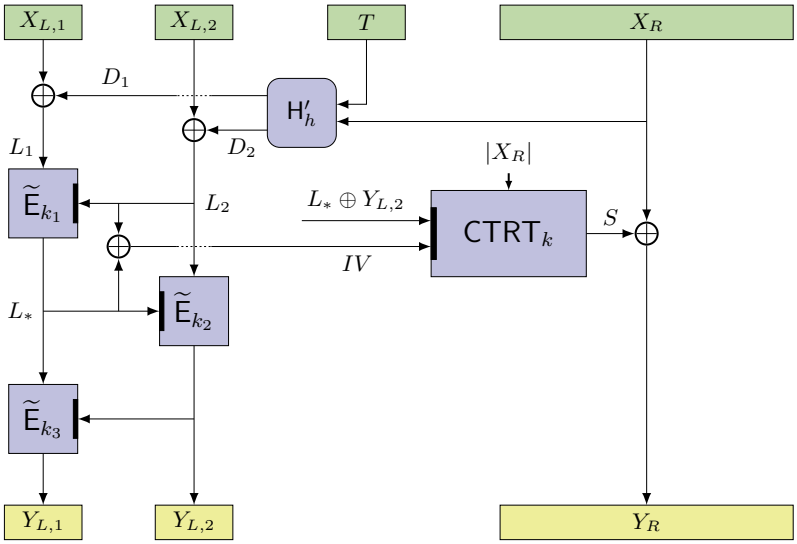


Figure 5.1: Graphical representation of the DE-THEC enciphering algorithm.

DE-THEC.$\text{En}_K(T, X_L, X_R)$	DE-THEC.$\text{De}_K(T, Y_L, Y_R)$	Proc. $\mathsf{CTRT}_k(W, IV, m)$
$(X_{L,1}, X_{L,2}) \leftarrow X_L$	$(Y_{L,1}, Y_{L,2}) \leftarrow Y_L$	$\kappa \leftarrow \lceil m/n \rceil$
$(k, k_1, k_2, k_3, h) \leftarrow K$	$(k, k_1, k_2, k_3, h) \leftarrow K$	**for** ctr $= 0$ to $\kappa - 1$
$(D_1, D_2) \leftarrow \mathsf{H}'_h(T, X_R)$	$L_* \leftarrow \widetilde{\mathsf{E}}^{-1}_{k_3}(Y_{L,2}, Y_{L,1})$	$\quad S_{\text{ctr}} \leftarrow \widetilde{\mathsf{E}}_k(IV \oplus \langle \text{ctr} \rangle_2, W)$
$L_1 \leftarrow X_{L,1} \oplus D_1$; $L_2 \leftarrow X_{L,2} \oplus D_2$	$L_2 \leftarrow \widetilde{\mathsf{E}}^{-1}_{k_2}(L_*, Y_{L,2})$	$S \leftarrow (S_0, \ldots, S_{\kappa-1})$
$L_* \leftarrow \widetilde{\mathsf{E}}_{k_1}(L_2, L_1)$	$L_1 \leftarrow \widetilde{\mathsf{E}}^{-1}_{k_1}(L_2, L_*)$	**return** $\lfloor S \rfloor_m$
$Y_{L,2} \leftarrow \widetilde{\mathsf{E}}_{k_2}(L_*, L_2)$	$IV \leftarrow L_* \oplus L_2$; $W \leftarrow L_* \oplus Y_{L,2}$	
$Y_{L,1} \leftarrow \widetilde{\mathsf{E}}_{k_3}(Y_{L,2}, L_*)$	$S \leftarrow \mathsf{CTRT}_k(W, IV, \lvert Y_R \rvert)$	
$Y_L \leftarrow (Y_{L,1}, Y_{L,2})$	$X_R \leftarrow Y_R \oplus S$	
$IV \leftarrow L_* \oplus L_2$; $W \leftarrow L_* \oplus Y_{L,2}$	$(D_1, D_2) \leftarrow \mathsf{H}'_h(T, X_R)$	
$S \leftarrow \mathsf{CTRT}_k(W, IV, \lvert X_R \rvert)$	$X_{L,1} \leftarrow L_1 \oplus D_1$; $X_{L,2} \leftarrow L_2 \oplus D_2$	
$Y_R \leftarrow X_R \oplus S$	$X_L \leftarrow (X_{L,1}, X_{L,2})$	
return (Y_L, Y_R)	**return** (X_L, X_R)	

Figure 5.2: Pseudocode description of the DE-THEC construction.

5.2 Why Domain Extension to Achieve Optimal Security?

We have seen that the left part of THEC construction is n-bit in size. Although it can offer full n-bit security, it comes with the condition that all the queried tweaks must be distinct. On the other hand, the left part of DE-THEC construction is $2n$-bits in size and can offer full n-bit security irrespective of the number of tweak repetitions. Now, for the THEC construction, we note that the tweakable CTR mode used in the right part of the construction requires $2n$-bit entropy, which should be provided from the left side of the construction. Since the left side possesses only n-bit of entropy, it cannot generate $2n$-bit randomness. On the other hand, for the DE-THEC construction, the left side possesses $2n$-bit randomness, which is fed to the right part of the construction. This alleviates the problem of graceful security degradation concerning the number of tweak repetitions and ensures full n-bit security regardless of the number of tweak repetitions.

6 Proof of Theorem 3 (DE-THEC Construction)

We follow the same structure as used in the proof of Theorem 2. We replace the four independently keyed TBC of the DE-THEC construction with four random tweakable permutations $\widetilde{\mathsf{P}}, \widetilde{\mathsf{P}}_1, \widetilde{\mathsf{P}}_2$ and $\widetilde{\mathsf{P}}_3$ and call the resulting construction DE-THEC* at the cost of the additional security term $4\mathbf{Adv}^{\text{STPRP}}_{\widetilde{\mathsf{E}}}(\mathcal{B})$. We

continue by employing the H-Coefficient technique with the real world being $\mathsf{DE\text{-}THEC}^*$ and the ideal world being the pair of oracles $(\mathsf{PP}, \mathsf{PP}^{-1})$, where $\mathsf{PP} \leftarrow\!\$\, \mathsf{TP}(\mathcal{T}, \mathcal{X}_L \times \mathcal{X}_R)$. The transcript τ is structured as follows $\tau = (\tau', h)$, where τ' contains all En, De and Gu queries the adversary made during the interaction, while h represents the actual hash function key, which is sampled uniformly at random from the set of all hash keys in the ideal world.

6.1 Extended Transcript and Definition of Bad Transcript

We extend the transcript of each query with the intermediate value L_*. In the real world, the L_* value will be the real value appearing in the construction $\mathsf{DE\text{-}THEC}^*$. In the ideal world, the L_* value will be sampled, simulating the behavior occurring in the real world. In addition, we also need to extend the guess oracle queries with values (X_L, X_R). Therefore, an i-th guess oracle query transcript will be of the form $(T^i, Y^i_{L,1}, Y^i_{L,2}, Y^i_R, \boldsymbol{V}^i, \mathsf{r}^i, X^i_{L,1}, X^i_{L,2}, X^i_R, L^i_*)$, where $\boldsymbol{V}^i$ and r^i represent the set of guessed values and the response of the guess oracle, respectively. Now, we describe how to sample these intermediate variables in the ideal world. Before it, we define the following:

Definition 7. *A guess oracle query is said to be fresh, if its triple (T^i, Y^i_L, Y^i_R) was queried for the first time. A fresh guess oracle query $(T^j, Y^j_{L,1}, Y^j_{L,2}, Y^j_R, \boldsymbol{V}^j)$ is called* independent*, if there is no i-th encipher or decipher query, with $i \neq j$, or i-th guess query, with $i < j$, such that $Y^j_{L,1} = Y^i_{L,1}$ and $Y^j_{L,2} = Y^i_{L,2}$. A fresh guess oracle query that is not independent, is called a* dependent *guess oracle query.*

In the rest of the proof, we denote the number of independent guess oracle queries with $q_{\text{gu-ind}}$. Now, we describe the procedure of sampling the intermediate values in the ideal world as follows:

1. Let $\Pi_1[\cdot,\cdot]$ be a table which is initialized to an empty table.
2. i-th independent Gu query: Sample (X^i_L, X^i_R) according to the permutation $\mathsf{PP}^{-1}(T^i, \cdot, \cdot)$ and set $\mathsf{PP}(T^i, X^i_L, X^i_R)$ to (Y^i_L, Y^i_R).
3. i-th dependent Gu query: We find the longest query (say j-th) that the i-th query depends on. Then set
$$X^i_R \leftarrow \begin{cases} Y^i_R \oplus \lfloor Y^j_R \oplus X^j_R \rfloor_{|Y^i_R|}, \text{ if } |Y^i_R| \leq \left|Y^j_R\right| \\ Y^i_R \oplus \left((Y^j_R \oplus X^j_R) \| U^i\right) \text{ where } U^i \leftarrow\!\$\, \{0,1\}^{|Y^i_R|-|Y^j_R|}, \text{ otherwise} \end{cases}$$
$$X^i_{L,1} \leftarrow \mathsf{H}'_h[1](T^i, X^i_R) \oplus X^j_{L,1} \oplus \mathsf{H}'_h[1](T^j, X^j_R)$$
$$X^i_{L,2} \leftarrow \mathsf{H}'_h[2](T^i, X^i_R) \oplus X^j_{L,2} \oplus \mathsf{H}'_h[2](T^j, X^j_R).$$
4. i-th non-fresh Gu query: For every i-th non-fresh guess oracle query, we set (X^i_L, X^i_R) to (X^j_L, X^j_R) of a previous guess oracle query (i.e., $j < i$) that had the same triple (T^i, Y^i_L, Y^i_R) queried.
5. Finally, for the i-th query, if $\Pi_1[X^i_{L,2} \oplus D^i_2, X^i_{L,1} \oplus D^i_1]$ (for encipher) or $\Pi_3^{-1}[Y^i_{L,2}, Y^i_{L,1}]$ (for decipher) is not set, we sample a value from
$$\{0,1\}^n \setminus \mathsf{rng}(\Pi_1[X^i_{L,2} \oplus D^i_2, \cdot]) \text{ or } \{0,1\}^n \setminus \mathsf{rng}(\Pi_3^{-1}[Y^i_{L,2}, \cdot])$$
respectively, and set L^i_* to the sampled value.

Definition 8. *An attainable transcript $\tau = (\tau', h)$ is bad, if there exist two queries i and j, such that either of the following holds:*

[B1] $\exists i \in [q_{\text{en}} + q_{\text{de}} + q_{\text{gu-ind}}], j \in [q_{\text{en}}], i < j$: $L^i_* = L^j_*, Y^i_{L,2} = Y^j_{L,2}$.
[B2] $\exists i \in [q_{\text{en}} + q_{\text{de}} + q_{\text{gu-ind}}], j \in [q_{\text{de}}], i < j$: $L^i_* = L^j_*, X^i_{L,2} \oplus \mathsf{H}'_h[2](T^i, X^i_R) = X^j_{L,2} \oplus \mathsf{H}'_h[2](T^j, X^j_R)$.
[B3] $\exists i \in [q_{\text{en}} + q_{\text{de}} + q_{\text{gu-ind}}], j \in [q_{\text{en}}], i < j$: $Y^i_{L,2} = Y^j_{L,2}, Y^i_{L,1} = Y^j_{L,1}$.
[B4] $\exists i, j \in [q_{\text{en}} + q_{\text{de}} + q_{\text{gu-ind}}], i < j$:
$$\begin{cases} X^i_{L,1} \oplus \mathsf{H}'_h[1](T^i, X^i_R) = X^j_{L,1} \oplus \mathsf{H}'_h[1](T^j, X^j_R) \\ X^i_{L,2} \oplus \mathsf{H}'_h[2](T^i, X^i_R) = X^j_{L,2} \oplus \mathsf{H}'_h[2](T^j, X^j_R) \end{cases}$$

[B5] *Either* $(\exists i, j \in [q_{\text{en}} + q_{\text{de}} + q_{\text{gu-ind}}], i \neq j)$ *or* $(\exists i \in [q], j \in [q_{\text{gu-dep}}]$*, such that* j *does not depend on query* i*), and* $\exists \text{ctr}_i \in [\kappa_i], \text{ctr}_j \in [\kappa_j]$*:*

$$\begin{cases} L^i_* \oplus Y^i_{L,2} = L^j_* \oplus Y^j_{L,2} \\ X^i_{L,2} \oplus \mathsf{H}'_h[2](T^i, X^i_R) \oplus L^i_* \oplus \text{ctr}_i = X^j_{L,2} \oplus \mathsf{H}'_h[2](T^j, X^j_R) \oplus L^j_* \oplus \text{ctr}_j. \end{cases}$$

[B6] *Either* $(\exists i, j \in [q_{\text{en}} + q_{\text{de}} + q_{\text{gu-ind}}], i \neq j)$ *or* $(\exists i \in [q], j \in [q_{\text{gu-dep}}]$*, such that* j *does not depend on query* i*), and* $\exists \text{ctr}_i \in [\kappa_i], \text{ctr}_j \in [\kappa_j]$*:*

$$\begin{cases} X^i_R[\text{ctr}_i] \oplus Y^i_R[\text{ctr}_i] = X^j_R[\text{ctr}_j] \oplus Y^j_R[\text{ctr}_j] \\ X^i_{L,2} \oplus \mathsf{H}'_h[2](T^i, X^i_R) \oplus L^i_* \oplus \text{ctr}_i = X^j_{L,2} \oplus \mathsf{H}'_h[2](T^j, X^j_R) \oplus L^j_* \oplus \text{ctr}_j. \end{cases}$$

[B7] $\exists i \in [q_{\text{gu}}]$*:* $X^i_L \in \boldsymbol{V}^i$ *(i.e., a successful guess oracle query)*

Lemma 2. *Let* BadT *be the set of bad (defined as above) attainable transcripts. Then, it holds*

$$\Pr[\mathsf{X}_{\text{id}} \in \mathsf{BadT}] \leq \epsilon_{\text{bad}} \leq q^2\epsilon + \frac{2qq_{\text{en}}}{2^{2n-2}} + \frac{qq_{\text{de}}}{2^{2n-2}} + \frac{2\min\{q^2 l_{\text{m}}, \sigma^2\}}{2^{2n-2}} + q_{\text{gu}} v \left(2\epsilon + \frac{1}{2^{2n-2}}\right).$$

To prove Lemma 2 and bound the probabilities that the bad events happen, let τ be some attainable transcript.

Bounding [B1]: For the first equation, at most $(j-1)$ values of L_* have been sampled up to the j-th enciphering query. From the sampling procedure of L_* values we know that L^j_* is drawn from the set of at least $2^n - j + 1$ elements. Under the assumption $q \leq 2^{n-1}$, the first equation $L^i_* = L^j_*$ can be bounded by

$$\frac{1}{2^n - j + 1} \leq \frac{1}{2^n - q} \leq \frac{1}{2^{n-1}}$$

As for the second equation if we denote the length of j-th query with l_j, and under the assumption $q \leq 2^{n-1} \leq 2^{l_j - 1}$ we can bound the probability of $Y^i_{L,2} = Y^j_{L,2}$ with

$$\frac{2^{l_j - n}}{2^{l_j} - (j-1)} \leq \frac{2^{l_j - n}}{2^{l_j} - q} \leq \frac{2^{l_j - n}}{2^{l_j - 1}} = \frac{1}{2^{n-1}}.$$

Finally, summing over all suitable i and j, the total bound for the condition [B1] is

$$\Pr[\text{B1}] \leq \frac{qq_{\text{en}}}{2^{2n-2}}. \tag{13}$$

Bounding [B2]: The first equation can be bounded as in the condition [B1]. The second equation can be bounded by $1/2^{n-1}$ using the randomness of $X^j_{L,2}$. So, summing up over all i and j, we have

$$\Pr[\text{B2}] \leq \frac{qq_{\text{de}}}{2^{2n-2}}. \tag{14}$$

Bounding [B3]: Since $(Y^j_{L,1}, Y^j_{L,2})$ has been sampled after i-th query, we can bound each of the equations with $\frac{1}{2^{n-1}}$. Summing over all suitable i and j, the total bound for the condition [B3] is

$$\Pr[\text{B3}] \leq \frac{qq_{\text{en}}}{2^{2n-2}}. \tag{15}$$

Bounding [B4]: If $(T^i, X^i_R) \neq (T^j, X^j_R)$, then the equations can be bounded by ϵ-AXU property of H'. Otherwise $(T^i, X^i_R) = (T^j, X^j_R)$, but in that case we know it must either hold $X^i_{L,1} \neq X^j_{L,1}$ or $X^i_{L,2} \neq X^j_{L,2}$ since otherwise query j would not be a valid one. Therefore, if $(T^i, X^i_R) = (T^j, X^j_R)$, it cannot happen that both equations are true at the same time. Summing up over all i and j, we have

$$\Pr[\text{B4}] \leq q^2\epsilon. \tag{16}$$

Bounding [B5]: There are two possibilities for choices of i and j in the event [B5], either (i) $\exists i, j \in [q_{\text{en}} + q_{\text{de}} + q_{\text{gu-ind}}], i \neq j$, or (ii) $\exists i \in [q], j \in [q_{\text{gu-dep}}]$, such that j does not depend on query i. The probability in both instances is calculated analogously, thus in the following we give explicitly the calculation for the instance (i), but sum over i, j taking both instances into account.

Assume wlog. that $i < j$. Now consider the following cases:

- $i \in [q_{\text{en}} + q_{\text{de}} + q_{\text{gu-ind}}], j \in [q_{\text{en}}]$: In this case the first equation is bounded by $1/2^{n-1}$ using the randomness of $Y^j_{L,2}$ and the second equation is bounded by $1/2^{n-1}$ due to the randomness of L^j_*.
- $i \in [q_{\text{en}} + q_{\text{de}} + q_{\text{gu-ind}}], j \in [q_{\text{de}} + q_{\text{gu-ind}}]$: In this case the first equation is bounded by $1/2^{n-1}$ due to the randomness of L^j_* and the second equation is bounded by $1/2^{n-1}$ using the randomness of $X^j_{L,2}$.

For fixed indices i and j, the number of choices for $\text{ctr}_i \oplus \text{ctr}_j$ is l_{m}. Alternatively, we can also count over the number of (i, ctr_i) and (j, ctr_j) pairs. Thus, by summing up over all i, j, ctr_i and ctr_j, the total bound for the condition [B5] is

$$\Pr[\text{B5}] \leq \frac{\min\{q^2 l_{\text{m}}, \sigma^2\}}{2^{2n-2}}. \tag{17}$$

Bounding [B6]: Similarly to the event [B5], there are two possibilities for choices of i and j in the event [B6]. The probability in both instances is calculated analogously, thus in the following we give explicitly the calculation for the first instance, but sum over i, j taking both instances into account.

Assume wlog. that $i < j$. The first equation can be bounded by the randomness of $X^j_R[\text{ctr}_j]$ if j was a decipher or independent guess query, or by the randomness of $Y^j_R[\text{ctr}_j]$ if j was an encipher query. The second equation can be bounded as in the condition [B5]. Summing up over all i, j, ctr_i and ctr_j, the total bound for the condition [B6] is

$$\Pr[\text{B6}] \leq \frac{\min\{q^2 l_{\text{m}}, \sigma^2\}}{2^{2n-2}}. \tag{18}$$

Bounding [B7]: To calculate the probability of a successful guess query, we split all guess oracle queries into equivalence classes. Two guess oracle queries are in the same equivalence class iff the queried quadruple $(T, Y_{L,1}, Y_{L,2}, Y_R)$ is the same. Consequently, an equivalence class will contain all guess queries with some fixed triple $(T, Y_{L,1}, Y_{L,2}, Y_R)$. Let q'_{gu} be the number of such equivalence classes and let q^c_{gu} denote the number of queries in c-th equivalence class. It holds that $1 \leq q^c_{\text{gu}} \leq q_{\text{gu}}$. Now, the first element in every equivalence class represents the first time the guess oracle was queried with the corresponding $(T, Y_{L,1}, Y_{L,2}, Y_R)$. We continue the calculation by differentiating two subcases.

- *First query was an **independent** guess oracle query.* This means the pair $(X_{L,1}, X_{L,2}, X_R)$ corresponding to the triple $(T, Y_{L,1}, Y_{L,2}, Y_R)$ was sampled according to the sampling described in Sec. 6.1. To calculate the probability of a successful guess query, we collect all guesses the adversary made for queries in the c-th equivalence class. Then, we use the principle of deferred decision to calculate the probability that sampled pair $(X_{L,1}, X_{L,2})$ is equal to one of the guesses by "delaying" the sampling of $(X_{L,1}, X_{L,2}, X_R)$ to the time the last guess query in this equivalence class was made. We can do that since, although the adversary's choices of guess sets $\boldsymbol{V}$ can be influenced during the game by other queries the adversary is making, the sampling of $(X_{L,1}, X_{L,2}, X_R)$ can be deferred to a later point with a side-effect of sample space being smaller. To conclude, for c-th equivalence class, the probability that $\mathcal{A}$ made a successful guess query, assuming $q \leq 2^{n-1}$, can be bounded by

$$\frac{q^c_{\text{gu}} v\, 2^{l_i - 2n}}{2^{l_i} - (i-1)} \leq \frac{q^c_{\text{gu}} v\, 2^{l_i - 2n}}{2^{l_i} - q} \leq \frac{q^c_{\text{gu}} v\, 2^{l_i - 2n}}{2^{l_i - 1}} = \frac{q^c_{\text{gu}} v}{2^{2n-1}},$$

  where i-th query is the last query in equivalence class c, and l_i denotes the length of the queries in c-th equivalence class.

- *First query was a **dependent** guess oracle query.* This means the pair $(X_{L,1},\ X_{L,2})$ corresponding to $(T, Y_{L,1}, Y_{L,2}, Y_R)$ was already set. To calculate the probability of a successful guess query in this subcase, we fix some guess value $(X^*_{L,1}, X^*_{L,2})$ from the guess set in a first guess oracle query in the respective equivalence class. We then calculate the probability that

$$X^*_{L,1} = \mathsf{H}'_h[1](T, X_R) \oplus X^j_{L,1} \oplus \mathsf{H}'_h[1](T^j, X^j_R)$$

and

$$X^*_{L,2} = \mathsf{H}'_h[1](T, X_R) \oplus X^j_{L,2} \oplus \mathsf{H}'_h[2](T^j, X^j_R)$$

  and then sum over all guess values in the first query and in the rest of the queries of the equivalence class. Let us call the event that the equations above hold CORRG. In addition, we define several more events: DEP_1 is an event that j-th query is a *previous* encipher, decipher or guess oracle query, DEP_2

is an event that j-th query is a *later* encipher query and DEP_3 is an event that j-th query is a *later* decipher query. Utilizing the law of total probability, we have that

$$\begin{aligned}\Pr[\mathrm{CORRG}] &= \sum_{i=1}^{3} \Pr[\mathrm{CORRG} \mid \mathrm{DEP}_i] \Pr[\mathrm{DEP}_i] \\ &\leq \sum_{i\in\{1,3\}} \Pr[\mathrm{CORRG} \mid \mathrm{DEP}_i] + \Pr[\mathrm{DEP}_2].\end{aligned}$$

Now we calculate the bound of each term independently.

- $\Pr[\mathrm{CORRG} \mid \mathrm{DEP}_1]$. If either $(T, X_R) \neq (T^j, X_R^j)$, then the term can be bounded by ϵ-AXU property of the hash function. Otherwise, the equations reduce to $X^*_{L,1} = X^j_{L,1}$ and $X^*_{L,2} = X^j_{L,2}$. The probability that those two reduced equations hold, given that $T = T^j, Y_{L,1} = Y^j_{L,1}, Y_{L,2} = Y^j_{L,2}$ and $Y_R = Y^j_R$, is 0. For j-th query being an encipher or decipher oracle query, such a guess oracle query would be forbidden by the game. If j-th query were a guess oracle query, the query $(T, Y_{L,1}, Y_{L,2}, Y_R)$ would not be a dependent oracle query, but a repeated query, which is a contradiction as well. So, we can bound $\Pr[\mathrm{CORRG} \mid \mathrm{DEP}_1]$ by ϵ.
- $\Pr[\mathrm{DEP}_2]$. This is equal to the probability that $Y_{L,1} = Y^j_{L,1}$ and $Y_{L,2} = Y^j_{L,2}$. We know this probability is upper bounded by $\frac{1}{2^{2n-2}}$, where the calculation follows the same pattern as it is in case [B4].
- $\Pr[\mathrm{CORRG} \mid \mathrm{DEP}_3]$. This term can be bounded by ϵ, with the calculation being analogous to the calculation for the event $\Pr[\mathrm{CORRG} \mid \mathrm{DEP}_1]$.

Now we can sum the Pr[CORRG] over all guesses and queries in c-th equivalence class and bound the probability of a successful guess query in c-th equivalence class in this subcase with

$$q^c_{\mathrm{gu}} v \left(2\epsilon + \frac{1}{2^{2n-2}}\right).$$

Summing the Pr[CORRG] over *all* guess queries and respective equivalence classes, we get

$$\Pr[\mathrm{B7}] \leq \sum_{i=1}^{q'_{\mathrm{gu}}} q^c_{\mathrm{gu}} v \left(2\epsilon + \frac{1}{2^{2n-2}}\right) = q_{\mathrm{gu}} v \left(2\epsilon + \frac{1}{2^{2n-2}}\right). \tag{19}$$

Summing up the bounds in Eqn. (13)-(19) we get

$$\epsilon_{\mathrm{bad}} \leq q^2\epsilon + \frac{2q q_{\mathrm{en}}}{2^{2n-2}} + \frac{q q_{\mathrm{de}}}{2^{2n-2}} + \frac{2\min\{q^2 l_{\mathrm{m}}, \sigma^2\}}{2^{2n-2}} + q_{\mathrm{gu}} v \left(2\epsilon + \frac{1}{2^{2n-2}}\right).$$

6.2 Analysis of Good Transcripts

Now we lower bound the ratio of the real to ideal interpolation probability for a good transcript. We begin by introducing a few notations needed for calculating the ratio.

- $\mathcal{T}_1$, $\mathcal{T}_2$ and $\mathcal{T}_3$ denote the set of distinct tweaks queried to $\widetilde{\mathsf{P}}_1, \widetilde{\mathsf{P}}_2$ and $\widetilde{\mathsf{P}}_3$. Let $\mathcal{T}_1^{\mathrm{en}}$ and $\mathcal{T}_3^{\mathrm{en}}$ denote the set of distinct tweaks queried to $\widetilde{\mathsf{P}}_1$ and $\widetilde{\mathsf{P}}_3$ during *encipher* queries. Similarly, let $\mathcal{T}_1^{\mathrm{dg}}$ and $\mathcal{T}_3^{\mathrm{dg}}$ denote the set of distinct tweaks queried to $\widetilde{\mathsf{P}}_3$ during *decipher* and *guess* oracle queries.
- $\mathcal{T}_{\mathrm{ctr}}$ denotes the set of distinct tweaks queried to $\widetilde{\mathsf{P}}$ in the counter mode in En, De, independent Gu queries, and calls to $\widetilde{\mathsf{P}}$ that generate the "independent"/new part of the key stream in *dependent* guess queries that are longer than the queries they depend on. Additionally, let σ_{d} be the total number of such "independent" key stream blocks for dependent oracle queries longer than the queries they depend on.
- α_T (resp. α_T^{en}, and α_T^{dg}), $T \in \mathcal{T}_1$ (resp. $\mathcal{T}_1^{\mathrm{en}}$, and $\mathcal{T}_1^{\mathrm{dg}}$), denotes the number of calls to $\widetilde{\mathsf{P}}_1$ (resp. during encipher queries, and decipher and guess queries) with tweak T.
- β_T, $T \in \mathcal{T}_2$ denotes the number of calls to $\widetilde{\mathsf{P}}_2$ with tweak T.
- γ_T (resp. γ_T^{dg}, and γ_T^{en}), $T \in \mathcal{T}_3$ (resp. $\mathcal{T}_3^{\mathrm{dg}}$, and $\mathcal{T}_3^{\mathrm{en}}$), denotes the number of calls to $\widetilde{\mathsf{P}}_3$ (resp. during decipher and guess, and encipher queries) with tweak T.
- θ_T, $T \in \mathcal{T}_{\mathrm{ctr}}$, denotes the number of calls to $\widetilde{\mathsf{P}}$ with tweak T.

- L denotes the number of distinct input lengths to $\mathsf{DE\text{-}THEC}^*$ for En, De and independent Gu queries. We denote the input lengths with $l_1, \ldots, l_L$. Let $l_{\min} = \min\{l_1, l_2, \ldots, l_L\}$
- c_i denotes the number of En, De and independent Gu queries of length l_i, while C_i denotes the number of distinct tweaks queried for inputs of length l_i.
- $c_{i,j}$ denotes the number of times j-th distinct tweak (out of C_i tweaks) has been queried (in En, De or independent Gu query) for inputs of length l_i.

Without loss of generality, we assume that each l_i is a multiple of n. Thus, we assume that each input length l_i consists of κ_i blocks, i.e., $l_i = \kappa_i n$. Moreover, $l_{\min} \geq 2n$. Therefore, it holds that

$$\sum_{T\in\mathcal{T}_1} \alpha_T = \sum_{T\in\mathcal{T}_2} \beta_T = \sum_{T\in\mathcal{T}_3} \gamma_T \leq q, \sum_{T\in\mathcal{T}_{\mathrm{ctr}}} \theta_T \leq \left(\sum_{i=1}^{L} c_i\kappa_i + \sigma_{\mathrm{d}}\right) - 2q, c_i = \sum_{j=1}^{C_i} c_{i,j} \tag{20}$$

Note that the interpolation probability for dependent guess oracle queries, that have shorter or equal inputs than the query they depend on, will always be one in both worlds as the values in such dependent guess oracle queries in an attainable transcript are already determined by the query they depend on and the sampled hash keys. We take care of the rest of the dependent guess oracle queries implicitly with σ_{d} variable.

Interpolation Probability in the Ideal World. The interpolation probability in the ideal world is

$$\Pr[X_i = \tau] = \frac{1}{|\mathcal{H}|} \times \prod_{i=1}^{L}\prod_{j=1}^{C_i} \frac{1}{(2^{l_i})_{c_{i,j}}} \times \frac{1}{2^{n\sigma_{\mathrm{d}}}} \times \prod_{T\in\mathcal{T}_1^{\mathrm{en}}} \frac{1}{(2^n)_{\alpha_T^{\mathrm{en}}}} \times \prod_{T\in\mathcal{T}_3^{\mathrm{dg}}} \frac{1}{(2^n)_{\gamma_T^{\mathrm{dg}}}},$$

where the second term corresponds to the sampling of responses to En, De and independent Gu oracle queries, the third term corresponds to the sampling of "independent" key stream for longer[10] dependent guess oracle queries and the last two terms interpolate the probability of L_* values occurring in the transcript.

Interpolation Probability in the Real World. As for the interpolation in the real world, we know that none of the bad conditions will occur in a good transcript τ. The conditions [B1]-[B6] ensure there will be no tweak-input or tweak-output collisions in any of the $\widetilde{\mathsf{P}}$, $\widetilde{\mathsf{P}}_1, \widetilde{\mathsf{P}}_2$ and $\widetilde{\mathsf{P}}_3$ calls.

Therefore, we can easily interpolate the probability of a good transcript τ in the real world as follows.

$$\Pr[X_r = \tau] = \frac{1}{|\mathcal{H}|} \times \prod_{T\in\mathcal{T}_1} \frac{1}{(2^n)_{\alpha_T}} \times \prod_{T\in\mathcal{T}_2} \frac{1}{(2^n)_{\beta_T}} \times \prod_{T\in\mathcal{T}_3} \frac{1}{(2^n)_{\gamma_T}} \times \prod_{T\in\mathcal{T}_{\mathrm{ctr}}} \frac{1}{(2^n)_{\theta_T}}.$$

Ratio of Real to Ideal Interpolation Probability.

$$\frac{\Pr[X_r = \tau]}{\Pr[X_i = \tau]} = \frac{\prod_{i=1}^{L}\prod_{j=1}^{C_i} (2^{l_i})_{c_{i,j}} \times 2^{n\sigma_{\mathrm{d}}}}{\prod_{T\in\mathcal{T}_1^{\mathrm{dg}}} (2^n)_{\alpha_T^{\mathrm{dg}}} \times \prod_{T\in\mathcal{T}_2} (2^n)_{\beta_T} \times \prod_{T\in\mathcal{T}_3^{\mathrm{en}}} (2^n)_{\gamma_T^{\mathrm{en}}} \times \prod_{T\in\mathcal{T}_{\mathrm{ctr}}} (2^n)_{\theta_T}} \tag{21}$$

By doing a similar algebraic calculation as was done for the good transcript analysis in the proof of THEC and by applying the facts $(2^n)_x \leq (2^n)^x$, $\sum_{T\in\mathcal{T}_1^{\mathrm{dg}}} \alpha_T^{\mathrm{dg}} + \sum_{T\in\mathcal{T}_3^{\mathrm{en}}} \gamma_T^{\mathrm{en}} \leq q$, $\sum_{i=1}^{L}\sum_{j=1}^{C_i} c_{i,j} \leq q$ and

[10] I.e., dependent guess oracle queries that are longer than the queries they depend on.

Eqn. (20), we have

$$\begin{aligned}
\frac{\Pr[X_r = \tau]}{\Pr[X_i = \tau]} &= \frac{\prod_{i=1}^{L}\prod_{j=1}^{C_i}\left(2^{l_i}\right)_{c_{i,j}} \times 2^{n\sigma_{\mathrm{d}}}}{\prod_{T\in\mathcal{T}_1^{\mathrm{dg}}}(2^n)_{\alpha_T^{\mathrm{dg}}} \times \prod_{T\in\mathcal{T}_2}(2^n)_{\beta_T} \times \prod_{T\in\mathcal{T}_3^{\mathrm{en}}}(2^n)_{\gamma_T^{\mathrm{en}}} \times \prod_{T\in\mathcal{T}_{\mathrm{ctr}}}(2^n)_{\theta_T}} \\
&\geq \frac{\prod_{i=1}^{L}\prod_{j=1}^{C_i}\left(2^{n\kappa_i}\right)_{c_{i,j}} \times 2^{n\sigma_{\mathrm{d}}}}{(2^n)^{\sum_{T\in\mathcal{T}_1^{\mathrm{dg}}}\alpha_T^{\mathrm{dg}}} \times (2^n)^{\sum_{T\in\mathcal{T}_2}\beta_T} \times (2^n)^{\sum_{T\in\mathcal{T}_3^{\mathrm{en}}}\gamma_T^{\mathrm{en}}} \times (2^n)^{\sum_{T\in\mathcal{T}_{\mathrm{ctr}}}\theta_T}} \\
&\geq \frac{\prod_{i=1}^{L}\prod_{j=1}^{C_i}\left(2^{n\kappa_i}\right)_{c_{i,j}} \times 2^{n\sigma_{\mathrm{d}}}}{2^{2nq} \times (2^n)^{\sum_{i=1}^{L}\sum_{j=1}^{C_i} c_{i,j}\kappa_i} \times (2^n)^{\sigma_{\mathrm{d}}} \times 2^{-2nq}} = \frac{\prod_{i=1}^{L}\prod_{j=1}^{C_i}\left(2^{n\kappa_i}\right)_{c_{i,j}}}{(2^n)^{\sum_{i=1}^{L}\sum_{j=1}^{C_i} c_{i,j}\kappa_i}} \\
&= \prod_{i=1}^{L}\prod_{j=1}^{C_i}\prod_{r=0}^{c_{i,j}-1}\left(1-\frac{r}{2^{n\kappa_i}}\right) \geq 1 - \sum_{i=1}^{L}\sum_{j=1}^{C_i}\sum_{r=0}^{c_{i,j}-1}\frac{r}{2^{n\kappa_i}} \geq 1 - \sum_{i=1}^{L}\sum_{j=1}^{C_i}\frac{c_{i,j}^2}{2^{n\kappa_i}} \\
&\geq 1 - \sum_{i=1}^{L}\sum_{j=1}^{C_i}\frac{q^2}{2^{n\kappa_{\min}}} \geq 1 - \underbrace{\frac{q^2}{2^{2n}}}_{\epsilon_{\mathrm{good}}}.
\end{aligned}$$

By combining Theorem 1, Lemma 2 and ϵ_{good}, one achieves the bound from the theorem statement. □

7 Instantiations and High-Security AEAD

We now examine the AEAD schemes resulting from our two RPRP constructions when combined with the EtE and EtD transforms [20]. We start by discussing the graceful security degradation of THEC and show that in several practical settings like TLS, SSH, and QUIC, the limitation on tweak repetitions from Theorem 2 is not detrimental due to the limits imposed by these protocols on the maximum number of decryption failures. We then propose instantiations of our THEC and DE-THEC constructions, and explain how one can construct MRAE and RUPAE schemes out of them. After that, we compare the concrete security bounds of AE schemes based on our schemes against other state-of-the-art MRAE and RUPAE schemes for various parameter sets as typically used in (D)TLS or as general AEAD schemes (without further restrictions). Finally, we talk about practical aspects of our schemes and compare them with AE schemes that achieve similar security levels.

7.1 Graceful Security Degradation in Practice

As noted already, the security of THEC is strictly beyond-birthday-bound only as long as the tweaks do not repeat, but security still degrades gracefully as the number of tweak repetitions increases. This property is not unique to THEC and is present in other BBB constructions such as the Accordion mode candidate Docked-Double-Decker-AES [23]. This poses a problem when applying the encode-then-encipher paradigm to obtain an AEAD scheme because the nonce is typically placed in the tweak, and while nonce should not repeat during encryption queries there is no restriction on nonce repetitions across decryption queries. This is a natural source of criticism for such constructions as the resulting AEAD scheme will not be secure beyond the birthday bound. However, in contrast to the AEAD security model, several secure-channel protocols like TLS, SSH, and QUIC only tolerate a limited number of decryption failures which implicitly limits the number of nonce repetitions during decryption. In particular, TCP-based channels like TLS and SSH allow only a *single* decryption failure, after which the connection is torn down. In UDP-based secure channels like DTLS and QUIC, a limited number of decryption failures is allowed, and the number depends on the AEAD scheme. For example, the QUIC RFC specifies a limit of $q_{\mathrm{D}} = 2^{21.5}$ decryption failures for CCM and $q_{\mathrm{D}} = 2^{36}$ for ChaCha20-Poly1305 ciphersuites [42, Section 6.6]. The DTLS RFC sets the limit to $q_{\mathrm{D}} = 2^{36}$ for GCM and ChaCha20-Poly1305 ciphersuites, while for CCM ciphersuite with short tags, the limit is set to $q_{\mathrm{D}} = 2^7$ [39]. Finally, the Bluetooth Core protocol also limits the maximum number of decryption failures to $q_{\mathrm{D}} = 3$ [25, Section 9.5].

In light of the above, we note that constructions with graceful degradation like THEC can still yield secure channels with very high security in several practical scenarios. On the other hand, we emphasize that this aspect is specific to THEC and does not apply to DE-THEC. That is, both EtE[DE-THEC] and EtD[DE-THEC] are fully beyond-birthday-bound secure as generic AEAD schemes without any further restriction on the number of decryption queries.

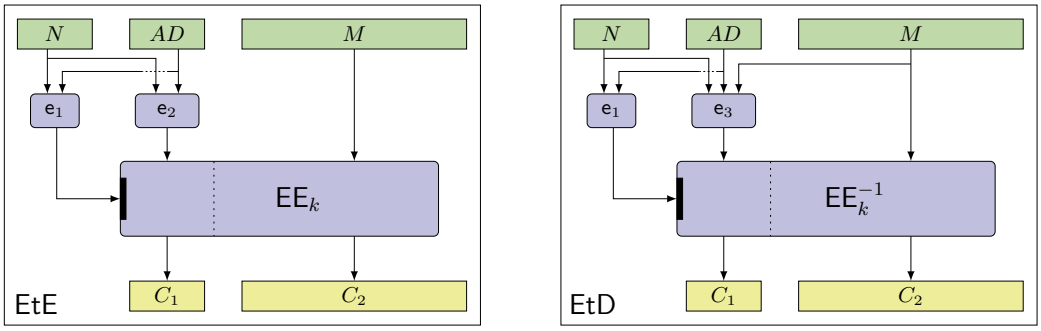


Figure 7.1: Graphical representation of encryption algorithm of EtE (**left**) and EtD (**right**) transforms.

7.2 Choice of Underlying Primitives

For the best security, we choose to instantiate the universal hash function H with a design whose security bound is independent of the input length.

One option is to use classical universal hash based on a polynomial construction, augmented with the 'trick' by Nyberg, Gilbert, and Robshaw [36] used in GCM-SST [11]. This approach requires computing a polynomial over a larger field, say $\mathrm{GF}(2^{174}-3)$, as described in [19]. The output is then processed using an AXU polynomial hash function over $\mathrm{GF}(2^{128})$ to produce the final tag. The resulting security bound will be of the form $\frac{c\ell}{2^{174}} + \frac{1}{2^{128}}$, for some constant c and input length ℓ. This bound is the second term when the input length is restricted to the range $\ell < \frac{2^{174-128}}{c}$. Another possibility for H is a TBC-based hash function with a length-independent bound.

To instantiate the $2n$-bit hash function H′ in DE-THEC, we have two options: 1. Use the concatenation of two independently-keyed instances of the polynomial hash function just described, or 2. Use ZHASH+ AXU hash function introduced by Datta et al. [17, 18].

As for the tweakable block cipher, Deoxys-TBC [31] and SKINNY [8], with 128-bit block sizes and 128-bit keys and tweaks appear to be the best choices. Both have been present for a longer time and have been subject to extensive cryptanalytic scrutiny. Between the two, Deoxys-TBC is a better choice due to its performance advantage from being implementable with AES-NI instructions.

7.3 AEAD Security and SOTA Comparison

Degabriele and Karadžić [20] have revisited the EtE and EtD paradigm and proposed transforms to turn an RPRP cipher into misuse-resistant AEAD and RUP-secure AEAD respectively. In essence, they both encode the AEAD inputs (N, H, M) as inputs to the RPRP cipher; but whereas the EtE transform uses the enciphering algorithm to encrypt, EtD uses the deciphering algorithm instead. Graphical representations of the encryption procedure for each scheme are shown in Figure 7.1, and the full pseudocode descriptions can be found in Appendix B. In these transforms, e_1 can be any injective function, e_2 is any deterministic function (e.g., $\mathsf{e}_2(X) := 0^n$), and e_3 is any function with non-repeating outputs—which is readily achieved by including the nonce in the output.

The security of the AEAD schemes obtained by applying the EtE and EtD transforms to THEC and DE-THEC is formally stated in Theorem 4, Theorem 5, and Theorem 6, where EtE[THEC] and EtE[DE-THEC] are misuse-resistant and EtD[DE-THEC] is secure against the release of unverified plaintext. In deriving these bounds, we assumed for THEC the instantiation of H we just described, for which ϵ, ρ and δ are all equal to $1/2^n$. For DE-THEC and corresponding hash function H′, we have ϵ equals to $1/2^{2n}$, when the maximum message length in blocks is limited to $l_M \leq 2^{45}$. In addition, in all three theorems σ represents the total number of message blocks queried across encryption and decryption queries, whereas μ represents the bound of nonce repetitions across encryption and decryption queries. Finally, to obtain BBB security, in Theorem 4 we require the right domain of THEC to be $\{0,1\}^{\geq n}$, whereas this was not a requirement for the RPRP security of THEC. As such, the EtE[THEC] scheme requires padding messages shorter than n bits.

Theorem 4. *Let* EtE[THEC] *be the nonce-based AEAD scheme realized from* THEC *over the domain* $\{0,1\}^n \times \{0,1\}^{\geq n}$. *Then for any adversary* $\mathcal{A}$ *making* q_{E} *encryption queries and* q_{D} *decryption queries, that repeats any nonce to the encryption oracle at most* μ_{E} *times, there exists an adversary* $\mathcal{B}$ *and it*

Use Case	$(q_{\mathrm{E}}, q_{\mathrm{D}}, l_M, \mu_{\mathrm{E}})$	Schemes							
		AES-GCM-SIV	XOCB	Deoxys-AE2	EtE[THEC]	DENC2	eGCM-SIV	ZAE	EtE[DE-THEC]
TLS Channel	$(2^{32}, 1, 2^{15}, 1)$	2^{-64}	2^{-66}	2^{-76}	2^{-80}	2^{-68}	2^{-76}	2^{-142}	2^{-174}
	$(2^{50}, 1, 2^{15}, 1)$	2^{-46}	2^{-46}	2^{-58}	2^{-62}	2^{-50}	2^{-58}	2^{-115}	2^{-138}
	$(2^{50}, 1, 2^{15}, 2^{12})$	2^{-37}	1	2^{-46}	2^{-50}	2^{-50}	2^{-58}	2^{-115}	2^{-138}
	$(2^{50}, 1, 2^{15}, 2^{50})$	1	1	2^{-8}	2^{-12}	2^{-50}	2^{-58}	2^{-115}	2^{-138}
DTLS Channel	$(2^{32}, 2^{6}, 2^{15}, 1)$	2^{-55}	2^{-66}	2^{-76}	2^{-80}	2^{-68}	2^{-76}	2^{-142}	2^{-174}
	$(2^{32}, 2^{6}, 2^{15}, 2^{12})$	2^{-55}	1	2^{-64}	2^{-68}	2^{-68}	2^{-76}	2^{-142}	2^{-174}
	$(2^{25}, 2^{23}, 2^{15}, 2^{25})$	2^{-49}	1	2^{-58}	2^{-62}	2^{-75}	2^{-83}	2^{-152}	2^{-188}
	$(2^{50}, 2^{36}, 2^{15}, 2^{50})$	1	1	2^{-8}	2^{-12}	2^{-50}	2^{-58}	2^{-115}	2^{-138}
AEAD	$(2^{16}, 2^{16}, 2^{20}, 1)$	2^{-41}	2^{-71}	2^{-86}	2^{-90}	2^{-78}	2^{-86}	2^{-165}	2^{-199}
	$(2^{32}, 2^{32}, 2^{20}, 1)$	1	2^{-55}	2^{-70}	2^{-74}	2^{-62}	2^{-70}	2^{-141}	2^{-167}
	$(2^{40}, 2^{40}, 2^{30}, 2^{12})$	1	1	2^{-40}	2^{-44}	2^{-44}	2^{-52}	2^{-112}	2^{-141}
	$(2^{50}, 2^{50}, 2^{20}, 2^{50})$	1	1	2^{-2}	2^{-6}	2^{-44}	2^{-52}	2^{-112}	2^{-131}

Table 1: Comparison of concrete (MR)AE security bounds for specific parameter choices and adversarial resources. If $\mu_{\mathrm{E}} = 1$, the table shows AE security level, otherwise it shows MRAE security level. Note that in the above we exclude the STPRP advantage of the underlying TBC.

holds

$$\mathbf{Adv}^{\mathrm{MRAE}}_{\mathsf{EtE[THEC]}}(\mathcal{A}) \leq \mathbf{Adv}^{\mathrm{STPRP}}_{\widetilde{\mathsf{E}}}(\mathcal{B}) + \frac{q_{\mathrm{E}}^2}{2^{2n+1}} + \frac{q(\mu-1)(l_M+3)}{2^{n-1}} \\ \frac{(\mu+2)\sigma'}{2^{n-1}} + \frac{q^2+q\sigma'}{2^{2n-3}} + \frac{2\min\{q^2 l_M, \sigma^2\}}{2^{2n-2}} + \frac{3q_{\mathrm{D}}}{2^{n-1}},$$

where $q = q_{\mathrm{E}} + q_{\mathrm{D}}$, $\sigma' = \sigma + q$ *and* $\mu = \max\{q_{\mathrm{D}}, \mu_{\mathrm{E}}\}$.

Theorem 5. *Let* EtE[DE-THEC] *be the nonce-based AEAD scheme realized from* DE-THEC *over the domain* $\{0,1\}^{2n} \times \{0,1\}^*$. *Then for any adversary* $\mathcal{A}$ *making* q_{E} *encryption queries and* q_{D} *decryption queries, that repeats any nonce to the encryption oracle at most* μ_{E} *times, there exists an adversary* $\mathcal{B}$ *and it holds*

$$\mathbf{Adv}^{\mathrm{MRAE}}_{\mathsf{EtE[DE\text{-}THEC]}}(\mathcal{A}) \leq 4\mathbf{Adv}^{\mathrm{STPRP}}_{\widetilde{\mathsf{E}}}(\mathcal{B}) + \frac{q_{\mathrm{E}}^2}{2^{2n+1}} + \frac{q^2}{2^{2n-4}} + \frac{\min\{q^2 l_M, \sigma^2\}}{2^{2n-3}} + \frac{q_{\mathrm{D}}}{2^{2n-3}},$$

where $q = q_{\mathrm{E}} + q_{\mathrm{D}}$.

Theorem 6. *Let* EtD[DE-THEC] *be the subtle AEAD scheme realized from* DE-THEC *over the domain* $\{0,1\}^{2n} \times \{0,1\}^*$. *Then there exists a leakage simulator* S, *such that for any adversary* $\mathcal{A}$ *making* $q_{\mathrm{E}} \leq 2^{2n-1}$ *encryption queries,* q_{D} *decryption queries and* q_{L} *queries to the leakage oracle, there exists an adversary* $\mathcal{B}$ *and it holds*

$$\mathbf{Adv}^{\mathrm{RUPAE}}_{\mathsf{EtD[DE\text{-}THEC]}}(\mathcal{A}) \leq 8\mathbf{Adv}^{\mathrm{STPRP}}_{\widetilde{\mathsf{E}}}(\mathcal{B}) + \frac{(q_{\mathrm{E}}+1)(q_{\mathrm{D}}+q_{\mathrm{L}})}{2^{2n-1}} + \frac{6q^2}{2^{2n-3}} + \frac{q_{\mathrm{E}}(q_{\mathrm{D}}+q_{\mathrm{L}})}{2^{2n}} + \frac{\min\{q^2 l_M, \sigma^2\}}{2^{2n-3}},$$

where $q = q_{\mathrm{E}} + q_{\mathrm{D}} + q_{\mathrm{L}}$.

As noted already, the EtE and EtD transforms have corresponding nonce-hiding variants. The differences between the conventional and nonce-hiding variants lie exclusively in the encoding functions and where the encoded redundancy is placed (in left X_L or right X_R part). Security bounds for the nonce-hiding variants can be derived in a similar manner. A notable difference is that for EtE[THEC], the μ parameter in the RPRP bound would only be bounded by AEAD adversary's parameter q_{D}, since in the nonce-hiding variant the nonce is not embedded into the tweak.

7.3.1 Security Comparison with SOTA Here, we compare concrete security levels of our AEAD schemes and other state-of-the-art AEAD schemes. For this security comparison, we selected the adversarial resources according to the typical use cases discussed in Subsection 7.1. In the first use case we

Use Case	(q_E, q_D, q_L, l_M)	Schemes		
		GCM-RUP	RIV	EtD[DE-THEC]
DTLS Channel	$(2^{16}, 2^{16}, 2^{16}, 2^{20})$	2^{-52}	2^{-69}	2^{-197}
	$(2^{24}, 2^{24}, 2^{24}, 2^{32})$	2^{-12}	2^{-41}	2^{-169}
	$(2^{32}, 2^{32}, 2^{32}, 2^{20})$	2^{-20}	2^{-37}	2^{-165}
AEAD	$(2^{40}, 2^{40}, 2^{40}, 2^{32})$	1	2^{-9}	2^{-137}
	$(2^{64}, 2^{64}, 2^{64}, 2^{20})$	1	1	2^{-101}

Table 2: Comparison of concrete RUPAE security bounds for specific parameter choices and adversarial resources: q_E, q_D, q_L are bounds on the number of queries to the respective oracles, and l_M is the message length in blocks. Note that in the above we exclude the STPRP advantage of the underlying TBC.

consider AEAD as used in a secure channel like TLS which tolerates only a single decryption failure, thus $q_D = 1$. The second considers AEAD deployed in a secure channel like DTLS (or QUIC) where more decryption failures are tolerated but are still limited. In both use cases we consider the typical maximum message length allowed by these protocols. The third use case, considers a general AEAD scenario where there is no restriction on the maximum number of forgery attempts that the adversary can make.

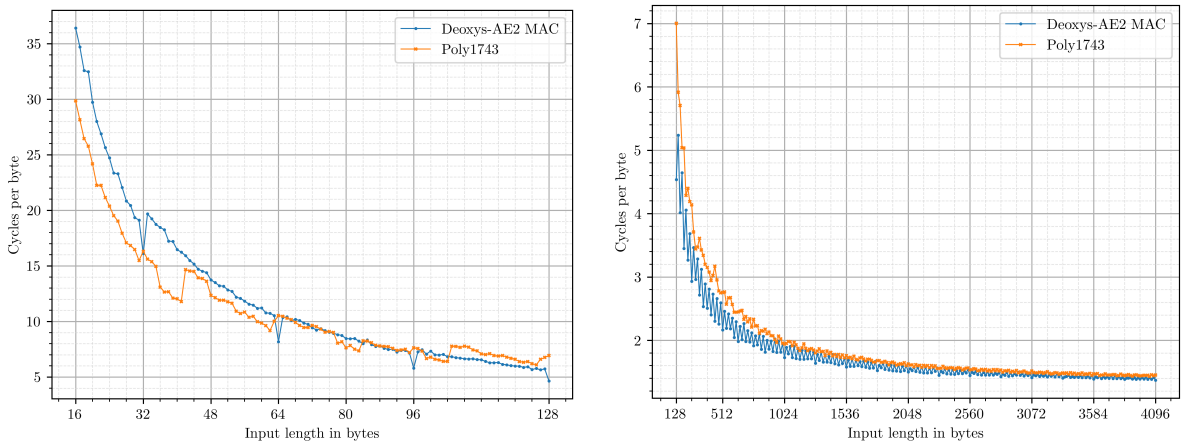
We first compare the MRAE security level of EtE[THEC] and EtE[DE-THEC] with AES-GCM-SIV [26], XOCB [5], the modern variant of CAESAR competition winner (for security in depth) Deoxys-AE2 [31], ZAE [30] and two very recent MRAE constructions DENC2 [14] and eGCM-SIV [15]. While XOCB is not MRAE-secure, it is one of the most recent BBB-secure AEAD proposals and is thus a good representative for the state-of-the-art in AEAD. As for ZAE and DENC2, they were shown to be a *deterministic AEAD* [40], but they can easily be transferred to a MRAE-secure scheme by including the nonce into associated data. The parameter μ_E represents nonce repetitions during encryption, and thus rows where $\mu_E = 1$ indicate standard AE security, whereas rows with $\mu_E > 1$ represent MRAE security. We derive the security levels of AES-GCM-SIV, XOCB, Deoxys-AE2, ZAE, DENC2 and eGCM-SIV through the security bounds described in their respective references. For deriving DENC2 and eGCM-SIV levels, we took scheme parameters the authors used for their implementations. The results are shown in Table 1. We observe that in all use cases EtE[THEC] attains a similar security level as Deoxys-AE2 (exceeding it by 4 bits), although, as we show later, it only requires half the number of TBC evaluations. On the other hand, EtE[DE-THEC] attains a much higher security level in all use cases, and provides more security than ZAE, which also targets full n bits of security.

Analogously, we compare the RUPAE security level of EtD[DE-THEC] with GCM-RUP [4] and RIV [1] (using the authors' instantiation). The results are shown in Table 2, where EtD[DE-THEC] clearly outperforms the others by a large security margin. Note that the $2n$-bits expansion in EtD is a necessity imposed by the EtD transform in order to achieve BBB security. GCM-RUP requires a similar expansion without achieving BBB security, and even if the expansion in RIV was extended to $2n$-bits, its security level would not improve. A short discussion on this matter can be found in Appendix C.

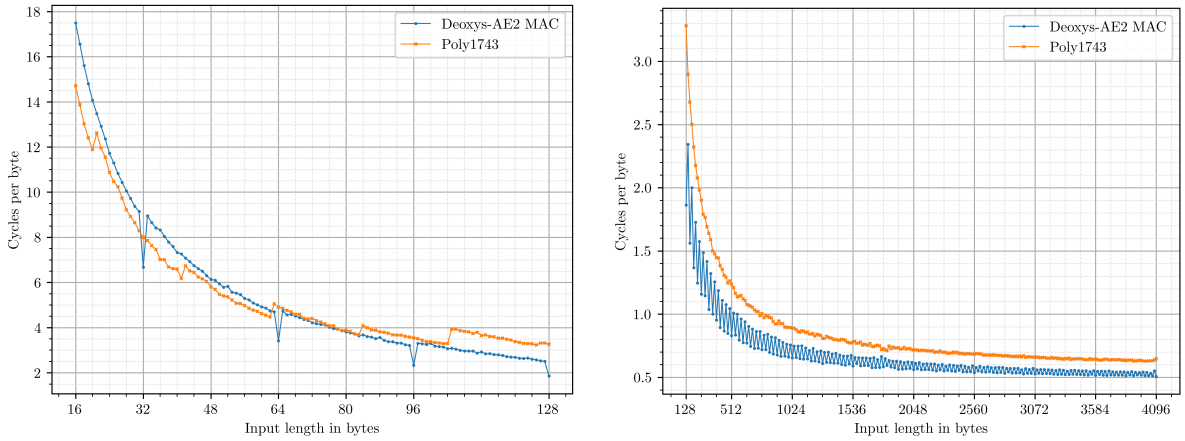
7.3.2 Partial Implementation Benchmarks and Performance Observations Finally, we proceed by looking at how proposed instantiations for hash functions H and H′ perform. For EtE[THEC] we compared the speed of Poly1743, the polynomial hash function based on $\mathrm{GF}(2^{174} - 3)$ described in Subsection 7.2, with the speed of MAC component of Deoxys-AE2. For Poly1743 we used the code that was automatically generated by the framework of Degabriele *et al.* [19], and for Deoxys-AE2 MAC component we used the code provided to us by Deoxys authors via private correspondence. As for the EtE[DE-THEC] and EtD[DE-THEC], we compared the speed of two independently keyed instances of the aforementioned polynomial hash function with the speed of ZHASH+ [17,18]. For ZHASH+ we used the optimized, non-vectorized, code provided by the authors on Github[11].

We measured the speed of these hash function (and MAC component of Deoxys-AE2) for different input lengths (in bytes), where the input is a concatenation of the message and associated data. Our benchmarks also implicitly include the encoding of message and associated data length. The results of

[11] https://github.com/ShibamCrS/HCTR-/tree/main/implementation/HCTR%2B_TLR3



(a) Benchmark on Intel Core i5-8265U CPU (Skylake microarchitecture).



(b) Benchmark on Intel Xeon Platinum 8488C CPU (Sapphire Rapids microarchitecture).

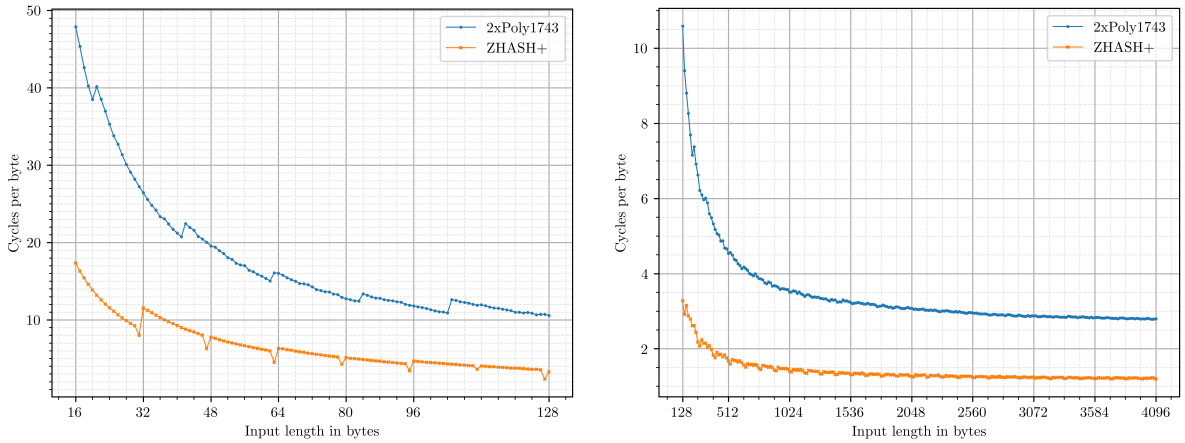
Figure 7.2: Benchmarks of Deoxys-AE2 MAC component and Poly1743-based hash function for different input lengths (in bytes). The plots on the left contain data points for every input length in $\{16, 17, \ldots, 128\}$, while plots on the right contain data points for every input length in $\{128, 144, \ldots, 4096\}$.

our measurements are located in Figure 7.2 and Figure 7.3, and the details about the benchmarking setup can be found in Appendix D.

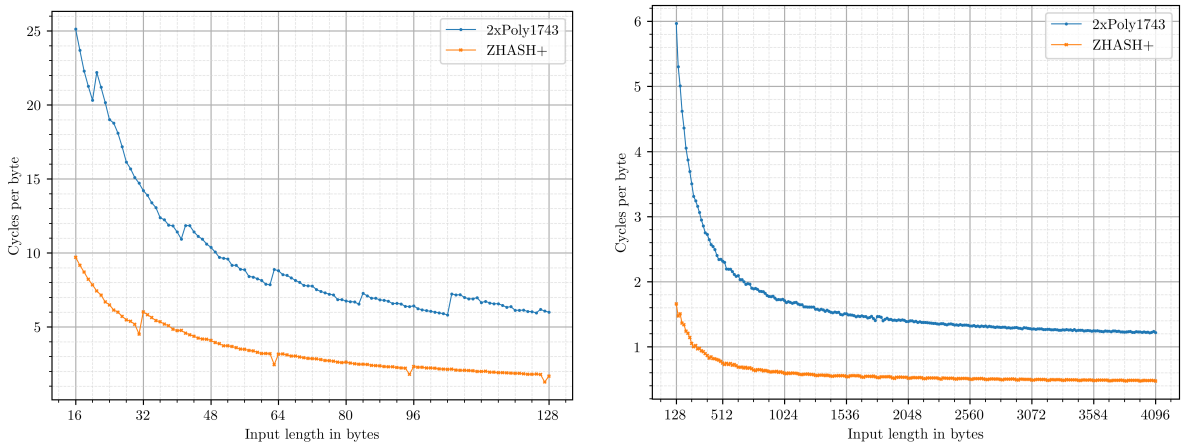
Reading the results in Figure 7.2, we make the following observations:

- Deoxys-AE2 MAC component slightly outperforms Poly1743 on longer messages. We expect that the TCTR mode in EtE[THEC], that uses the "lighter" Deoxys-TBC-128-256 version with 14 internal rounds, would outperform the TCTR mode in Deoxys-AE2, that uses the "heavier" Deoxys-TBC-128-384 version with 16 internal rounds[12]. Since the hash/MAC and counter mode layers are inherently sequential in both schemes, we expect encryption speed of EtE[THEC] instantiated with Poly1743 at least matches the speed of Deoxys-AE2 while offering 4 more bits of security. One should note, we use computer-generated code for Poly1743 and that code could be manually optimized.
- The two-pass decryption process in Deoxys-AE2 and EtE[THEC] can be done in parallel in both schemes. We believe that EtE[THEC] decryption would outperform Deoxys-AE2 decryption since the Poly1743 code works on integer arithmetic CPU instructions and would not clutter the AES-NI instruction pipeline as Deoxys-AE2.

[12] The inherently heavier tweakey schedule in Deoxys-TBC-128-384 does not play a role here since Deoxys-AE2 implementation precomputes the tweakey values for counter tweak inputs.



(a) Benchmark on Intel Core i5-8265U CPU (Skylake microarchitecture).



(b) Benchmark on Intel Xeon Platinum 8488C CPU (Sapphire Rapids microarchitecture).

Figure 7.3: Benchmarks of ZHASH+ and two independent instantiations of Poly1743-based hash function for different input lengths (in bytes). The plots on the left contain data points for every input length in $\{16, 17, \ldots, 128\}$, while plots on the right contain data points for every input length in $\{128, 144, \ldots, 4096\}$.

Looking at Figure 7.3 and potential instantiations of EtE[DE-THEC] and EtD[DE-THEC], we see that ZHASH+ outperforms two independent instantiations of Poly1743 by several factors. As such, ZHASH+ becomes the main candidate for H' instantiation.

Additionally, we can also compare the theoretical cost of ZAE and EtE[DE-THEC] since they use the same primitives, and as seen in Table 1, both stand out in terms of security from other listed schemes. ZAE has $\frac{a+m+2}{2}+m+4$ TBC calls and $3(a+m+2)/2$ GF(2^{128}) doubling operations, while EtE[DE-THEC] has $\frac{a+m+2}{2}+m+2$ TBC calls and the same number of GF(2^{128}) doubling operations[13]. Two TBC calls less in EtE[DE-THEC] come from more efficient finalization operation in ZHASH+ in comparison to ZMAC deployed in ZAE. Thus, we expect EtE[DE-THEC] to be slightly more efficient than ZAE, most notably for short inputs.

8 Conclusion

In this paper, we presented two RPRP constructions from tweakable block ciphers that achieve BBB security. While THEC provides BBB security up to a certain tweak multiplicity, DE-THEC achieves optimal security regardless of tweak repetitions. In turn, these constructions yield efficient AEAD schemes

[13] We did not count the two constant initialization TBC calls included in both ZMAC in ZAE, and ZHASH+ in our instantiation.

that are resilient to misuse and that can also process significantly larger amounts of data as required by today's cloud computing applications.

References

1. Abed, F., Forler, C., List, E., Lucks, S., Wenzel, J.: RIV for robust authenticated encryption. In: Peyrin, T. (ed.) FSE 2016. LNCS, vol. 9783, pp. 23–42. Springer, Berlin, Heidelberg (Mar 2016). https://doi.org/10.1007/978-3-662-52993-5_2
2. Andreeva, E., Bhati, A.S., Preneel, B., Vizár, D.: 1, 2, 3, fork: Counter mode variants based on a generalized forkcipher. IACR Trans. Symm. Cryptol. **2021**(3), 1–35 (2021). https://doi.org/10.46586/tosc.v2021.i3.1-35
3. Andreeva, E., Bogdanov, A., Luykx, A., Mennink, B., Mouha, N., Yasuda, K.: How to securely release unverified plaintext in authenticated encryption. In: Sarkar, P., Iwata, T. (eds.) ASIACRYPT 2014, Part I. LNCS, vol. 8873, pp. 105–125. Springer, Berlin, Heidelberg (Dec 2014). https://doi.org/10.1007/978-3-662-45611-8_6
4. Ashur, T., Dunkelman, O., Luykx, A.: Boosting authenticated encryption robustness with minimal modifications. In: Katz, J., Shacham, H. (eds.) CRYPTO 2017, Part III. LNCS, vol. 10403, pp. 3–33. Springer, Cham (Aug 2017). https://doi.org/10.1007/978-3-319-63697-9_1
5. Bao, Z., Hwang, S., Inoue, A., Lee, B., Lee, J., Minematsu, K.: XOCB: Beyond-birthday-bound secure authenticated encryption mode with rate-one computation. In: Hazay, C., Stam, M. (eds.) EUROCRYPT 2023, Part IV. LNCS, vol. 14007, pp. 532–561. Springer, Cham (Apr 2023). https://doi.org/10.1007/978-3-031-30634-1_18
6. Barwell, G., Martin, D.P., Oswald, E., Stam, M.: Authenticated encryption in the face of protocol and side channel leakage. In: Takagi, T., Peyrin, T. (eds.) ASIACRYPT 2017, Part I. LNCS, vol. 10624, pp. 693–723. Springer, Cham (Dec 2017). https://doi.org/10.1007/978-3-319-70694-8_24
7. Barwell, G., Page, D., Stam, M.: Rogue decryption failures: Reconciling AE robustness notions. In: Groth, J. (ed.) 15th IMA International Conference on Cryptography and Coding. LNCS, vol. 9496, pp. 94–111. Springer, Cham (Dec 2015). https://doi.org/10.1007/978-3-319-27239-9_6
8. Beierle, C., Jean, J., Kölbl, S., Leander, G., Moradi, A., Peyrin, T., Sasaki, Y., Sasdrich, P., Sim, S.M.: The SKINNY family of block ciphers and its low-latency variant MANTIS. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016, Part II. LNCS, vol. 9815, pp. 123–153. Springer, Berlin, Heidelberg (Aug 2016). https://doi.org/10.1007/978-3-662-53008-5_5
9. Bellare, M., Ng, R., Tackmann, B.: Nonces are noticed: AEAD revisited. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019, Part I. LNCS, vol. 11692, pp. 235–265. Springer, Cham (Aug 2019). https://doi.org/10.1007/978-3-030-26948-7_9
10. Bellare, M., Rogaway, P.: Encode-then-encipher encryption: How to exploit nonces or redundancy in plaintexts for efficient cryptography. In: Okamoto, T. (ed.) ASIACRYPT 2000. LNCS, vol. 1976, pp. 317–330. Springer, Berlin, Heidelberg (Dec 2000). https://doi.org/10.1007/3-540-44448-3_24
11. Campagna, M., Maximov, A., Mattsson, J.P.: Galois Counter Mode with secure short tags (GCM-SST). Internet Draft (Sep 2024), https://datatracker.ietf.org/doc/draft-mattsson-cfrg-aes-gcm-sst/
12. Chakraborty, D., Nandi, M.: An improved security bound for HCTR. In: Nyberg, K. (ed.) FSE 2008. LNCS, vol. 5086, pp. 289–302. Springer, Berlin, Heidelberg (Feb 2008). https://doi.org/10.1007/978-3-540-71039-4_18
13. Chen, S., Steinberger, J.P.: Tight security bounds for key-alternating ciphers. In: Advances in Cryptology - EUROCRYPT 2014,. pp. 327–350 (2014)
14. Chen, Y.L., Dutta, A., Jha, A., Nandi, M.: Towards optimally secure deterministic authenticated encryption schemes. In: Fehr, S., Fouque, P.A. (eds.) EUROCRYPT 2025, Part I. LNCS, vol. 15601, pp. 3–32. Springer, Cham (May 2025). https://doi.org/10.1007/978-3-031-91107-1_1
15. Chung, W., Hwang, S., Kim, S., Lee, B., Lee, J.: Making GCM great again: Toward full security and longer nonces. In: Fehr, S., Fouque, P.A. (eds.) EUROCRYPT 2025, Part I. LNCS, vol. 15601, pp. 33–61. Springer, Cham (May 2025). https://doi.org/10.1007/978-3-031-91107-1_2
16. Coron, J., Dodis, Y., Mandal, A., Seurin, Y.: A domain extender for the ideal cipher. In: Micciancio, D. (ed.) Theory of Cryptography, 7th Theory of Cryptography Conference, TCC 2010, Zurich, Switzerland, February 9-11, 2010. Proceedings. Lecture Notes in Computer Science, vol. 5978, pp. 273–289. Springer (2010). https://doi.org/10.1007/978-3-642-11799-2_17, https://doi.org/10.1007/978-3-642-11799-2_17
17. Datta, N., Dutta, A., Ghosh, S., List, E., Nandi, H.: HCTR+: An optimally secure TBC-based accordion mode. IACR Trans. Symm. Cryptol. **2025**(3), 183–229 (2024). https://doi.org/10.46586/tosc.v2025.i3.183-229
18. Datta, N., Dutta, A., Ghosh, S., List, E., Nandi, H.: HCTR+: An optimally secure TBC-based accordion mode. Cryptology ePrint Archive, Report 2024/2053 (2024), https://eprint.iacr.org/2024/2053
19. Degabriele, J.P., Gilcher, J., Govinden, J., Paterson, K.G.: SoK: Efficient design and implementation of polynomial hash functions over prime fields. In: 2024 IEEE Symposium on Security and Privacy. pp. 3128–3146. IEEE Computer Society Press (May 2024). https://doi.org/10.1109/SP54263.2024.00132

20. Degabriele, J.P., Karadžić, V.: Overloading the nonce: Rugged PRPs, nonce-set AEAD, and order-resilient channels. In: Dodis, Y., Shrimpton, T. (eds.) CRYPTO 2022, Part IV. LNCS, vol. 13510, pp. 264–295. Springer, Cham (Aug 2022). https://doi.org/10.1007/978-3-031-15985-5_10
21. Degabriele, J.P., Karadžić, V.: Populating the zoo of rugged pseudorandom permutations. In: Guo, J., Steinfeld, R. (eds.) ASIACRYPT 2023, Part VIII. LNCS, vol. 14445, pp. 270–300. Springer, Singapore (Dec 2023). https://doi.org/10.1007/978-981-99-8742-9_9
22. Degabriele, J.P., Melloni, A., Münch, J.P., Stam, M.: Counter galois onion (CGO) for tor: Fast non-malleable onion encryption. Cryptology ePrint Archive, Report 2025/583 (2025), https://eprint.iacr.org/2025/583
23. Dobraunig, C., Matusiewicz, K., Mennink, B., Tereschenko, A.: Efficient instances of docked double decker with AES, and application to authenticated encryption. Cryptology ePrint Archive, Report 2024/084 (2024), https://eprint.iacr.org/2024/084
24. Dutta, A., Nandi, M.: Tweakable HCTR: A BBB secure tweakable enciphering scheme. In: Chakraborty, D., Iwata, T. (eds.) INDOCRYPT 2018. LNCS, vol. 11356, pp. 47–69. Springer, Cham (Dec 2018). https://doi.org/10.1007/978-3-030-05378-9_3
25. Group, B.S.I.: Bluetooth Core Version 5.4 Specification, BR/EDR Controller, Part H, Security Specification, https://www.bluetooth.com/wp-content/uploads/Files/Specification/HTML/Core-54/out/en/br-edr-controller/security-specification.html, Accessed on 14.05.2025
26. Gueron, S., Langley, A., Lindell, Y.: AES-GCM-SIV: Specification and analysis. Cryptology ePrint Archive, Report 2017/168 (2017), https://eprint.iacr.org/2017/168
27. Hoang, V.T., Krovetz, T., Rogaway, P.: Robust authenticated-encryption AEZ and the problem that it solves. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015, Part I. LNCS, vol. 9056, pp. 15–44. Springer, Berlin, Heidelberg (Apr 2015). https://doi.org/10.1007/978-3-662-46800-5_2
28. Hoang, V.T., Reyhanitabar, R., Rogaway, P., Vizár, D.: Online authenticated-encryption and its nonce-reuse misuse-resistance. In: Gennaro, R., Robshaw, M.J.B. (eds.) CRYPTO 2015, Part I. LNCS, vol. 9215, pp. 493–517. Springer, Berlin, Heidelberg (Aug 2015). https://doi.org/10.1007/978-3-662-47989-6_24
29. Iwata, T., Khairallah, M., Minematsu, K., Peyrin, T.: Duel of the titans: The Romulus and Remus families of lightweight AEAD algorithms. IACR Trans. Symm. Cryptol. **2020**(1), 43–120 (2020). https://doi.org/10.13154/tosc.v2020.i1.43-120
30. Iwata, T., Minematsu, K., Peyrin, T., Seurin, Y.: ZMAC: A fast tweakable block cipher mode for highly secure message authentication. In: Katz, J., Shacham, H. (eds.) CRYPTO 2017, Part III. LNCS, vol. 10403, pp. 34–65. Springer, Cham (Aug 2017). https://doi.org/10.1007/978-3-319-63697-9_2
31. Jean, J., Nikolic, I., Peyrin, T., Seurin, Y.: The deoxys AEAD family. Journal of Cryptology **34**(3), 31 (Jul 2021). https://doi.org/10.1007/s00145-021-09397-w
32. Khairallah, M.: A note on 'tweakable HCTR: A BBB secure tweakable enciphering scheme'. Cryptology ePrint Archive, Report 2024/600 (2024), https://eprint.iacr.org/2024/600
33. Luby, M., Rackoff, C.: How to construct pseudorandom permutations from pseudorandom functions. SIAM Journal on Computing **17**(2) (1988)
34. Mattewson, N.: Tor developers mailing list: Proposal 359: Counter Galois Onion, Updated, https://lists.torproject.org/mailman3/hyperkitty/list/tor-dev@lists.torproject.org/thread/TQNPG3FMGFADHFCKK3FOEYBH3YK4AOI2/, Accessed on 18.08.2025
35. NIST: Requirements for Cryptographic Accordions. National Institute of Standards and Technology (NIST) (Apr 2025), https://doi.org/10.6028/NIST.IR.8552. Accessed on 18.08.2025
36. Nyber, K., Gilbert, H., Robshaw, M.: Galois MAC with forgery probability close to ideal (Jun 2005), https://csrc.nist.gov/CSRC/media/Projects/Block-Cipher-Techniques/documents/BCM/Comments/general-comments/papers/Nyberg_Gilbert_and_Robshaw.pdf. Accessed on 14.05.2025
37. Patarin, J.: The "Coefficients H" technique. In: Avanzi, R.M., Keliher, L., Sica, F. (eds.) Selected Areas in Cryptography, 15th International Workshop, SAC 2008, Sackville, New Brunswick, Canada, August 14-15, Revised Selected Papers. Lecture Notes in Computer Science, vol. 5381, pp. 328–345. Springer (2008). https://doi.org/10.1007/978-3-642-04159-4_21, https://doi.org/10.1007/978-3-642-04159-4_21
38. Peyrin, T., Seurin, Y.: Counter-in-tweak: Authenticated encryption modes for tweakable block ciphers. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016, Part I. LNCS, vol. 9814, pp. 33–63. Springer, Berlin, Heidelberg (Aug 2016). https://doi.org/10.1007/978-3-662-53018-4_2
39. Rescorla, E., Tschofenig, H., Modadugu, N.: The Datagram Transport Layer Security (DTLS) Protocol Version 1.3. RFC 9147 (Proposed Standard) (Apr 2022). https://doi.org/10.17487/RFC9147, https://www.rfc-editor.org/rfc/rfc9147.txt
40. Rogaway, P., Shrimpton, T.: A provable-security treatment of the key-wrap problem. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 373–390. Springer, Berlin, Heidelberg (May / Jun 2006). https://doi.org/10.1007/11761679_23
41. Shrimpton, T., Terashima, R.S.: A modular framework for building variable-input-length tweakable ciphers. In: Sako, K., Sarkar, P. (eds.) ASIACRYPT 2013, Part I. LNCS, vol. 8269, pp. 405–423. Springer, Berlin, Heidelberg (Dec 2013). https://doi.org/10.1007/978-3-642-42033-7_21
42. Thomson (Ed.), M., Turner (Ed.), S.: Using TLS to Secure QUIC. RFC 9001 (Proposed Standard) (May 2021). https://doi.org/10.17487/RFC9001, https://www.rfc-editor.org/rfc/rfc9001.txt

A Rugged Pseudorandom Permutation Game

The security game corresponding to the RPRP notion, introduced in Subsection 2.2, is given in Figure A.1.

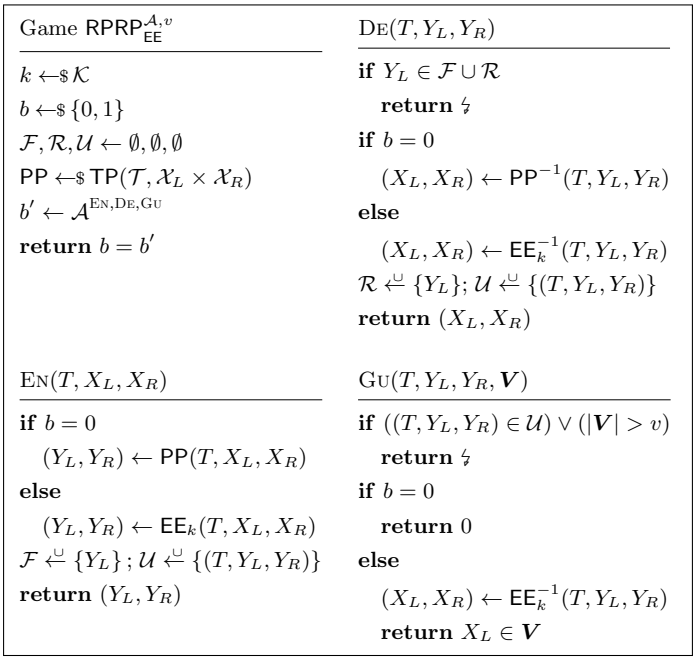


Figure A.1: The game used to define RPRP security for a tweakable cipher EE.

B Pseudocode and Security Theorems of EtE and EtD Transforms

The reference pseudocode of EtE and EtD transforms is given in Figure B.1.

[20, Theorem 2]. *Let* EtE *be the nonce-based AEAD scheme realized from a tweakable cipher* EE *over the domain* $\{0,1\}^{N_L} \times \{0,1\}^{\geq N_R}$*. Then for any adversary* $\mathcal{A}$ *making* q_{E} *encryption queries and* q_{V} *verification queries, there exists an adversary* $\mathcal{B}$ *such that*

$$\mathbf{Adv}^{\mathrm{MRAE}}_{\mathrm{EtE}}(\mathcal{A}) \leq \mathbf{Adv}^{\mathrm{RPRP}}_{\mathsf{EE}}(\mathcal{B}, 1) + \frac{q_{\mathrm{E}}^2}{2^{N_L+N_R+1}},$$

where $\mathcal{B}$ *makes* q_{E} *encipher queries and* q_{V} *guess queries.*

[20, Theorem 3]. *Let* EtD *be the subtle AEAD scheme realized from a tweakable cipher* EE *over the domain* $\{0,1\}^{N_L} \times \{0,1\}^{\geq N_R}$*, and let* e_3 *be a* (δ, t)*-collision resistant deterministic function. Then there exists a leakage simulator* S*, such that for any adversary* $\mathcal{A}$ *making* $q_{\mathrm{E}} \leq 2^{N_L-1}$ *encryption queries,* q_{D} *decryption queries and* q_{L} *queries to the leakage oracle, there exist RPRP adversaries* $\mathcal{B}$ *and* $\mathcal{C}$ *such that*

$$\mathbf{Adv}^{\mathrm{RUPAE}}_{\mathrm{EtD}}(\mathcal{A}) \leq \mathbf{Adv}^{\mathrm{RPRP}}_{\mathsf{EE}}(\mathcal{B}, 1) + \mathbf{Adv}^{\mathrm{RPRP}}_{\mathsf{EE}}(\mathcal{C}, 1) + \delta + \frac{(q_{\mathrm{E}}+1)(q_{\mathrm{D}}+q_{\mathrm{L}})}{2^{N_L-1}} + \frac{(q_{\mathrm{E}}+q_{\mathrm{D}}+q_{\mathrm{L}})^2}{2^{N_L+N_R}} + \frac{q_{\mathrm{E}}(q_{\mathrm{D}}+q_{\mathrm{L}})}{2^{N_L}}.$$

The adversary $\mathcal{B}$ *makes* q_{E} *encipher queries and* $q_{\mathrm{D}} + q_{\mathrm{L}}$ *decipher queries. The adversary* $\mathcal{C}$ *makes at most* q_{E} *encipher and at most* $q_{\mathrm{D}} + q_{\mathrm{L}}$ *decipher queries.*

$\mathsf{EtE}[\mathsf{EE}].\mathsf{Enc}_k(N, AD, M)$	$\mathsf{EtE}[\mathsf{EE}].\mathsf{Dec}_k(N, AD, C)$
$T \leftarrow \mathsf{e}_1(N, AD)$	$(C_1, C_2) \leftarrow C$
$Z \leftarrow \mathsf{e}_2(N, AD)$	$T \leftarrow \mathsf{e}_1(N, AD)$
$(C_1, C_2) \leftarrow \mathsf{EE}_k(T, Z, M)$	$Z \leftarrow \mathsf{e}_2(N, AD)$
return (C_1, C_2)	$(Z', M') \leftarrow \mathsf{EE}^{-1}(T, C_1, C_2)$
	if $Z' = Z$: **return** M'
	else : **return** $\perp$

$\mathsf{EtD}[\mathsf{EE}].\mathsf{Enc}_k(N, AD, M)$	$\mathsf{EtD}[\mathsf{EE}].\mathsf{Dec}_k(N, AD, C)$	$\mathsf{EtD}[\mathsf{EE}].\mathsf{Leak}_k(N, AD, C)$
$T \leftarrow \mathsf{e}_1(N, AD)$	$(C_1, C_2) \leftarrow C$	$(C_1, C_2) \leftarrow C$
$Z \leftarrow \mathsf{e}_3(N, AD, M)$	$T \leftarrow \mathsf{e}_1(N, AD)$	$T \leftarrow \mathsf{e}_1(N, AD)$
$(C_1, C_2) \leftarrow \mathsf{EE}_k^{-1}(T, Z, M)$	$(Z', M') \leftarrow \mathsf{EE}_k(T, C_1, C_2)$	$(Z', M') \leftarrow \mathsf{EE}_k(T, C_1, C_2)$
return (C_1, C_2)	$Z \leftarrow \mathsf{e}_3(N, AD, M')$	$Z \leftarrow \mathsf{e}_3(N, AD, M')$
	if $Z' = Z$: **return** M'	**if** $Z' = Z$: **return** $\top$
	else : **return** $\perp$	**else** : **return** M'

Figure B.1: Pseudocode of the EtE (**top**) and EtD (**bottom**) transforms [20].

C Security of "Domain-Extended" GCM-RUP and RIV Variants

In GCM-RUP [4, Algorithm 1], doubling the tag size equals doubling the redundancy from 0^τ to $0^{2\tau}$ bits. That doubling does not influence the confidentiality security of GCM-RUP, which is reduced to plain CTR mode security. Thus, the confidentiality would still only be birthday-bound secure. Similarly, the RIV [1], that is, the proposed instantiation by its authors, uses (XOR-)CTR mode as one of the components, which suffers from the same problem of BB confidentiality security.

D Benchmarking Setup Details

For each input length, we start the benchmark by running the evaluated function 500 times to "warm-up" the cache, and then executed the function 1500 times and measured the average number of cycles per byte needed for one evaluation. We ran the experiment 40 times, discarded the six best and worst results, then calculated the average number of cycles per byte over the rest of 28 runs. The resulting averages are shown in Figure 7.2 and Figure 7.3. We ran the benchmarks on:

1. A notebook using Intel Core i5-8265U CPU (Skylake microarchitecture), with a 1.60 GHz base frequency and the hyper-threading, frequency scaling and turbo mode functionalities disabled.
2. A server using Intel Xeon Platinum 8488C CPU (Sapphire Rapids microarchitecture), with a 2.00 GHz base frequency and 3.80 GHz maximal turbo frequency. Here, the hyper-threading, frequency scaling and turbo mode functionalities were enabled since we did not have administrative rights to disable those options.