IACR Transactions on Symmetric Cryptology
ISSN 2519-173X, Vol. 2021, No. 2, pp. 446–468. DOI:10.46586/tosc.v2021.i2.446-468

Power Yoga: Variable-Stretch Security of CCM for Energy-Efficient Lightweight IoT

Emiljano Gjiriti[1], Reza Reyhanitabar[2] and Damian Vizár[3]

[1] École Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland, emiljano.gjiriti@epfl.ch
[2] Siemens Energy, Erlangen, Germany, reza.reyhanitabar@siemens-energy.com
[3] Swiss Center for Electronics and Microtechnology (CSEM), Neuchâtel, Switzerland, damian.vizar@csem.ch

Abstract. The currently ongoing NIST LWC project aims at identifying new standardization targets for lightweight authenticated encryption with associated data (AEAD) and (optionally) lightweight cryptographic hashing. NIST has deemed it important for performance and cost to be optimized on relevant platforms, especially for short messages. Reyhanitabar, Vaudenay and Vizár (Asiacrypt 2016) gave a formal treatment for security of nonce-based AEAD with variable stretch, i.e., when the length of the authentication tag is changed between encryptions without changing the key. They argued that AEAD supporting variable stretch is of practical interest for constrained applications, especially low-power devices operated by battery, due to the ability to flexibly trade communication overhead and level of integrity.
In this work, we investigate this hypothesis with affirmative results. We present vCCM, a variable-stretch variant of the standard CCM and prove it is secure when used with variable stretch. We then experimentally measure the energy consumption of a real-world wireless sensor node when encrypting and sending messages with vCCM and CCM, respectively. Our projections show that the flexible trade of integrity level and ciphertext expansion can lead up to 21% overall energy consumption reduction in certain scenarios. As vCCM is obtained from the widely-used CCM by a black-box transformation, allowing any existing CCM implementations to be reused as-is, our results can be immediately put to use in practice. vCCM is all the more relevant because neither the NIST LWC project, nor any of the candidates give a consideration for the support of variable stretch and the related integrity-overhead trade-off.

Keywords: Authenticated encryption · variable-length tags · lightweight crypto · provable security · AES-CCM

1 Introduction

In the absence of a broadly accepted precise definition, lightweight cryptography can be roughly understood to comprise cryptographic designs created and optimized for a specific design trade-off (along the axes of computational complexity, memory complexity, security level, qualitative security properties etc.), such that this trade-off is not well-served by the existing general-purpose cryptography. In practice, a vast majority of these specific design targets is dictated by severely constrained resources of the intended applications, such as limited amount of energy available in a battery-operated device, limited power available in externally-powered passive RFIDs, limited computational resources (HW area/RAM/ROM/computational cycles) available for an implementation of cryptography, or a limited bandwidth available for the communication overhead due to the use of cryptography [KM].

Received: 2021-03-01 Accepted: 2021-05-01 Published: 2021-06-11

NIST LWC. Even though the field has already seen nearly two decades of research activities [BP17, SWE02, WSRE03, nisa], lightweight cryptography has been truly brought into the spotlight only recently by the ongoing NIST Lightweight Cryptography (LWC) Standardization process [nisb]. NIST LWC aims at identifying the most suitable candidates for a standardization of two symmetric-key functionalities: authenticated encryption with associated data (AEAD) [Rog02], and optionally also cryptographic hashing. Mirroring our informal definition, NIST states as their motivation that "the majority of current cryptographic algorithms were designed for desktop/server environments, many of these algorithms do not fit into constrained devices" [nisa]. The design requirement naturally implied is that new special-purpose candidate algorithms, in particular lightweight AEAD schemes, should perform significantly better in constrained environments compared to current general-purpose NIST standards, such as GCM and CCM modes for AEAD (instantiated with standard primitives e.g. AES).

Among other criteria, NIST has designated several cost metrics (e.g., area, memory, energy consumption), performance metrics (e.g., latency, throughput, power consumption) and implementation flexibility (in terms of achieving cost/performance trade-offs) for the evaluation of a candidate algorithm's merit. As has been the case in the other standardization projects and academic competitions, the target security levels have been fixed (e.g., 112-bit or 224-bit security against key recovery [nisb]), and the design parameters of a competing algorithm, such as the nonce-length or the ciphertext expansion must be fixed, resulting in one or more candidate instances.

Security-overhead trade-off. Meeting a clearly communicated, and generally accepted quantitative security level is a baseline necessity in modern cryptography. However, insisting on a single monolithic quantitative target for several security properties (of an AEAD scheme), and the related fixing of certain design parameters make it impossible to achieve an optimal trade-off between security and overhead, which is one of the central principles in lightweight cryptography [BP17]. In particular, we argue that facilitating a "flexible" security-overhead trade-off can be a practically relevant feature for several use cases of lightweight algorithms, provided such a "flexible" trade-off can be done safely.

Variable stretch. To clarify what is meant by "flexible" security-overhead trade-off, consider for example a nonce-based AEAD algorithm that is secure to use with variable stretch [RVV16]; that is, under the same key, several tag lengths (a.k.a. ciphertext expansion or stretch) can be securely used for authenticated encryption of different messages. The tag length is a main determinant of the security level of an AEAD algorithm, but also a major factor in increasing communication overhead, hence the energy consumption in a battery-powered constrained device. The ease of varying the tag length without the need for re-keying thus allows the authentication-related communication overhead to be adapted to the level of sensitivity of each transmitted messages, having a significant impact on the operation of the device overall, when considering energy consumption as a critical cost metric in many application areas.

The impact of the incurred communication overhead can be quite significant in use cases where authenticated encryption of predominantly (very) short messages is intended, and unsurprisingly such use cases are abundant in practice [ALP+19, ADP+20]. For example, in applications such as smart parking lots, the payload to be sent by the sensors most of the time is just one bit ("free" or "occupied"), together with a unique identifier of the sensor in the parking lot (e.g., 2 bytes). Here, a high tag length, e.g. 128 bits, will then be the main contributor to the communication cost compared to the actually transmitted information, and arguably disproportional to the impact of a successful forgery (changing between "free" and "occupied" in a single status update). However, the sensors will also need to send and receive management traffic, which is likely going to be more sensitive (e.g., "go to sleep" or "enter high performance mode" commands being used for

DoS attacks), requiring a higher level of protection. We note that NIST LWC [nisb] has stressed as a design requirement that lightweight AEAD submissions "shall be optimized to be efficient for short messages (e.g., as short as 8 bytes).

Yet, such an objective has neither been made explicit in the NIST LWC nor, to the best of our knowledge, has any of the submitted AEAD candidates in NIST LWC yet addressed the problem explicitly or claimed how to benefit from a variable-stretch AEAD security for flexible trade-off between security and energy consumption.

Related work. The support of variable tag length in AEAD schemes has been investigated by Reyhanitabar, Vaudenay and Vizár [RVV16] from both a misuse-resistance viewpoint and optimizing computation and communication overheads in resource-constrained devices. The work formalizes the security of nonce-based AEAD schemes with variable stretch in the security notion nvae (for nonce-based variable-stretch AE), establishes relations with existing notions, and proposes how to achieve and prove nvae security with the help of the so-called kess property (for key-equivalent separation by stretch). An analogous treatment of variable tag length security of nonce-based MAC schemes as well as an investigation of possible construction strategies has been done by Ghosh and Sarkar [GS19]. Safavi-Naini, Lisý and Desmedt proposed a game-theoretic approach for determining economically optimal tag lengths for different types of messages in a single application [SLD17]. This result complements the previous ones, and provides additional arguments for the hypothesis that varying the tag length in an application is desirable.

Contribution. Our main motivation is to investigate usefulness of the notion of variable-stretch AEAD for enabling the security-overhead trade-off with respect to the energy consumption overhead due to the ciphertext expansion. In many applications relying on battery-operated low-power devices (such as wireless sensor networks, smart medical implants, etc.) energy is a critical resource and decreasing its consumption one of the major optimization targets.

Towards this goal, we first propose a variable-stretch variant of the CCM standard, naming it vCCM, and prove that it is a secure nonce-based variable-stretch AEAD scheme, i.e., nvAE-secure as formalized by Reyhanitabar et al. [RVV16]. We then experimentally measure the energy consumption of a real-world low-power wireless sensor platform in a simple model scenario when using vCCM and CCM, respectively. Based on our measurements, we estimate that the overall energy consumption can be reduced by about 8-21% in applications similar to our scenario, just by treating a majority of messages with a shorter stretch than used for the most sensitive messages (which is not possible with AEAD schemes that do not support the variable-stretch security).

Our results also raise the bar for evaluation of new (not only) lightweight algorithms, adding the criterion of supporting variable tag length for achieving optimal security-energy trade-off, with our lightweight variant of the existing CCM standard as a baseline reference.

Variable-stretch CCM in the wild. CCM mode for AEAD was originally proposed in 2002 [WHF02, WHF03] for inclusion in the IEEE 802.11i standard. Despite initial academic criticisms [RW03], CCM has become one of the most widely supported AEAD schemes in real-world crypto, being used in IPsec, TLS, Bluetooth Low Energy (BLE) and a minor variant (CCM*) in the ZigBee standard, to mention some.

Our proposed vCCM scheme is obtained from CCM using a black-box transformation, i.e., without requiring any changes in the internals of the standard CCM scheme. This property has been the primary design goal in this work: it allows practitioners to benefit from existing software and hardware implementations of CCM, while instantly enabling the trade-off between security level and energy consumption in a flexible and provably sound manner. At the same time, our black-box transform itself is very lightweight, requiring only a few elementary operations.

At the time of writing this paper, it is expected that NIST LWC project will conclude in a year, resulting hopefully in one or several promising lightweight AEAD schemes with significant improvements compared to current standards; however, considering the time for standardization, inclusion in protocols and actual deployment in embedded systems, the real availability of the new schemes can expected to be several years away at best. In the meantime, we believe that salvaging a widely implemented standard such as CCM and *instantly* enabling a provably graceful trading of security for energy savings is of high interest, with the potential to bring measurable improvements to real-world applications.

2 Preliminaries and Prior AE Definitions

Notations. We let $a \leftarrow\$ \mathcal{S}$ denote the uniform sampling of an element in a finite set $\mathcal{S}$ and assigning it to the variable a. All strings are binary strings. We let $|X|$ denote the length of a string X, and $X\|Y$ the concatenation of two strings X and Y. We let ε denote the empty string of length 0. We let $\{0,1\}^*$ denote the set of all strings of arbitrary finite lengths (s.t. $\varepsilon \in \{0,1\}^*$) and we let $\{0,1\}^n$ denote the set of all strings of length n for a positive integer n. For a string $X \in \{0,1\}^*$, we let $\mathsf{left}_\ell(X)$ denote the ℓ leftmost bits of X for an $0 \leq \ell \leq |X|$. We let $\mathbb{N}$ denote the set of all (positive) natural numbers and $\mathbb{N}_0 = \mathbb{N} \cup \{0\}$. As the specification of CCM is byte-oriented, we define the short-hands $\mathsf{B} = \{0,1\}^8$ for the set of bytes, and for a byte string $X \in \mathsf{B}^*$ the length in bytes $|X|_8 = |X|/8$. The sets of arbitrary length byte strings B^* and byte strings of m bytes B^m are defined in a natural way. It is also useful to define $(\{0,1\}^n)^*$, the set of all strings whose length is a multiple of some integer n.

Longest common blockwise prefix and prefix-freeness. For two strings $X, Y \in (\{0,1\}^n)^*$, we let $\mathsf{llcp}_n(X,Y) = \max_{0 \leq i \leq \min(|X|/n, |Y|/n)} \{i \mid \mathsf{left}_{i \cdot n}(X) = \mathsf{left}_{i \cdot n}(Y)\}$ denote the length of the longest common blockwise prefix of X and Y in n-bit blocks. We further let $\mathsf{llcp}_n(X_1, \ldots, X_m; Y) = \max\{\mathsf{llcp}_n(X_1, Y), \ldots, \mathsf{llcp}_N(X_m, Y)\}$ denote the maximal value of llcp_n between a string $Y \in (\{0,1\}^n)^*$ and a collection of m strings $X_1, \ldots, X_m \in (\{0,1\}^n)^*$.

We call a mapping $\mathsf{encode} : \mathcal{S} \to (\{0,1\}^n)^*$ that encodes elements of a domain $\mathcal{S}$ into strings of multiple-of-n-bit length *prefix-free* if for each pair of distinct $s, s' \in \mathcal{S}$ we have $\mathsf{llcp}_n(\mathsf{encode}(s), \mathsf{encode}(s')) < \min(|\mathsf{encode}(s)|, |\mathsf{encode}(s')|)/n$.

Resource-parameterized advantage. The (in)security of a scheme Π with respect to a notion **xxx** is measured by taking the maximum over all adversaries $\mathscr{A}$ which use resources bounded by **r**.

Blockciphers. A blockcipher is a function $E : \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ for an integer n, the blocksize, such that the mapping $E_K(\cdot) = E(K, \cdot)$ is a permutation of $\{0,1\}^n$ for every $K \in \mathcal{K}$ with some finite $\mathcal{K}$. We define the security of blockciphers though indistinguishability from a random permutation by an adversary $\mathscr{A}$ as:

$$\mathbf{Adv}^{\mathrm{prp}}_E(\mathscr{A}) = \Pr\left[K \leftarrow\$ \mathcal{K} : \mathscr{A}^{E_K} \Rightarrow 1\right] - \Pr\left[\pi \leftarrow\$ \mathrm{Perm}(n) : \mathscr{A}^{\pi} \Rightarrow 1\right]$$

where $\mathrm{Perm}(n)$ is the set of all the permutations over n-bit strings.

Similarly, we define for a function with the same signature $E : \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ and a finite $\mathcal{K}$ (not insisting on the permutation property of $E_K(\cdot)$)

$$\mathbf{Adv}^{\mathrm{prf}}_E(\mathscr{A}) = \Pr\left[K \leftarrow\$ \mathcal{K} : \mathscr{A}^{E_K} \Rightarrow 1\right] - \Pr\left[f \leftarrow\$ \mathrm{Func}(n) : \mathscr{A}^{\pi} \Rightarrow 1\right]$$

where $\mathrm{Func}(n)$ is the set of functions functions from $\{0,1\}^n$ to itself.

The resource parameterized advantage functions are defined with the time complexity (t) of the adversary and the total number of queries (q) asked by the adversary.

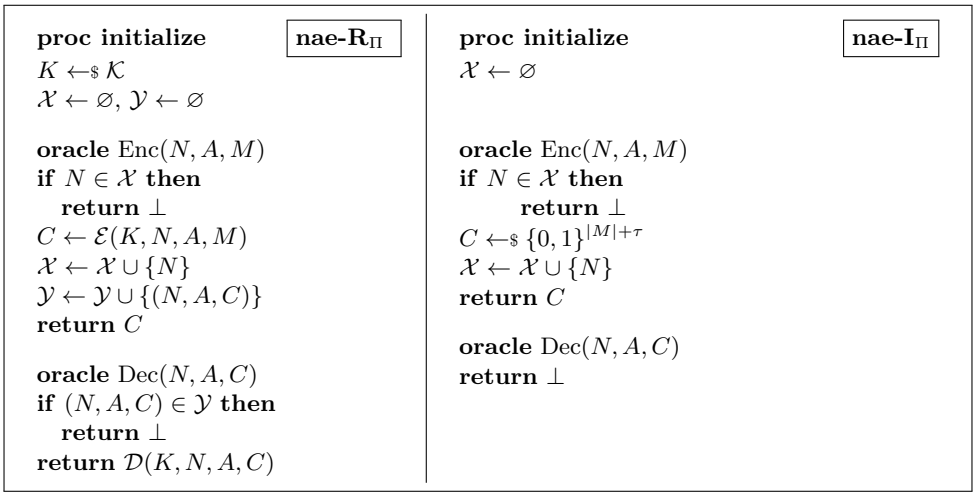


Figure 1: All-in-one definition of nAE security for a scheme $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ with ciphertext expansion τ.

We now recall the security notions for nonce-based AE (nAE) schemes with associated data (AEAD schemes) [Rog02].

Syntax. Following Reyhanitabar et al. [RVV16], we use the augmented syntax of AEAD schemes that includes a stretch variable. A scheme for authenticated encryption is a triplet $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ where $\mathcal{K} \subseteq \{0,1\}^*$ is the set of keys endowed with a (uniform) distribution and $\mathcal{E} : \mathcal{K} \times \mathcal{N} \times \mathcal{A} \times \mathcal{I}_T \times \mathcal{M} \to \mathcal{C}$ and $\mathcal{D} : \mathcal{K} \times \mathcal{N} \times \mathcal{A} \times \mathcal{I}_T \times \mathcal{C} \to \mathcal{M} \cup \{\perp\}$ are the encryption and decryption algorithm respectively, both deterministic and stateless. The argument domains of Π are $\mathcal{N}$ nonce space, $\mathcal{A}$ AD space, $\mathcal{M}$ plaintext space, $\mathcal{C}$ ciphertext space, and $\mathcal{I}_T$ stretch space (i.e. the set of ciphertext expansion values that can be applied upon encryption), and we require that $\mathcal{N} \subseteq \{0,1\}^*$, $\mathcal{M} \subseteq \{0,1\}^*$, $\mathcal{A} \subseteq \{0,1\}^*$, $\mathcal{C} \subseteq \{0,1\}^*$ and $\mathcal{I}_T \subseteq \mathbb{N}$.

We also require that if $M \in \mathcal{M}$ then $\{0,1\}^{|M|} \subseteq \mathcal{M}$, and that the scheme Π be correct and tidy [NRS14]. Correctness means that for every $(K, N, A, \tau, M) \in \mathcal{K} \times \mathcal{N} \times \mathcal{A} \times \mathcal{I}_T \times \mathcal{M}$, if $\mathcal{E}(K, N, A, \tau, M) = C$ then $\mathcal{D}(K, N, A, \tau, C) = M$. Tidiness means that for every $(K, N, A, \tau, C) \in \mathcal{K} \times \mathcal{N} \times \mathcal{A} \times \mathcal{I}_T \times \mathcal{C}$, if $\mathcal{D}(K, N, A, \tau, C) = M \neq \perp$ then $\mathcal{E}(K, N, A, \tau, M) = C$. In both cases $|C| = |M| + \tau$ where τ denotes the *stretch*.

An *ordinary* nAE scheme that has a fixed stretch τ can be seen as a special case of the just-defined syntax, having $\mathcal{I}_T = \{\tau\}$. The value of the stretch argument being trivially assigned in all encryption and decryption calls, it can be omitted from the list of arguments in both cases. We let $\Pi[\tau]$ denote an ordinary nonce-based AE scheme obtained from a nonce-based AE scheme with variable stretch Π by fixing the expansion value for all queries to some value $\tau \in \mathcal{I}_T$.

nAE security definition. The all-in-one nAE security definition by Rogaway and Shrimpton captures AE security as indistinguishability of the real encryption and decryption algorithms from a random strings oracle and an always-reject oracle in a nonce-respecting, chosen ciphertext attack. The **nae** advantage of an adversary $\mathscr{A}$ against a scheme Π is defined as $\mathbf{Adv}^{\mathbf{nae}}_{\Pi}(\mathscr{A}) = \Pr[\mathscr{A}^{\mathbf{nae\text{-}R}_\Pi} \Rightarrow 1] - \Pr[\mathscr{A}^{\mathbf{nae\text{-}I}_\Pi} \Rightarrow 1]$ with the security games defined in Figure 1.

nvAE Security Definition. Reyhanitabar et al. define the security of an nvAE scheme

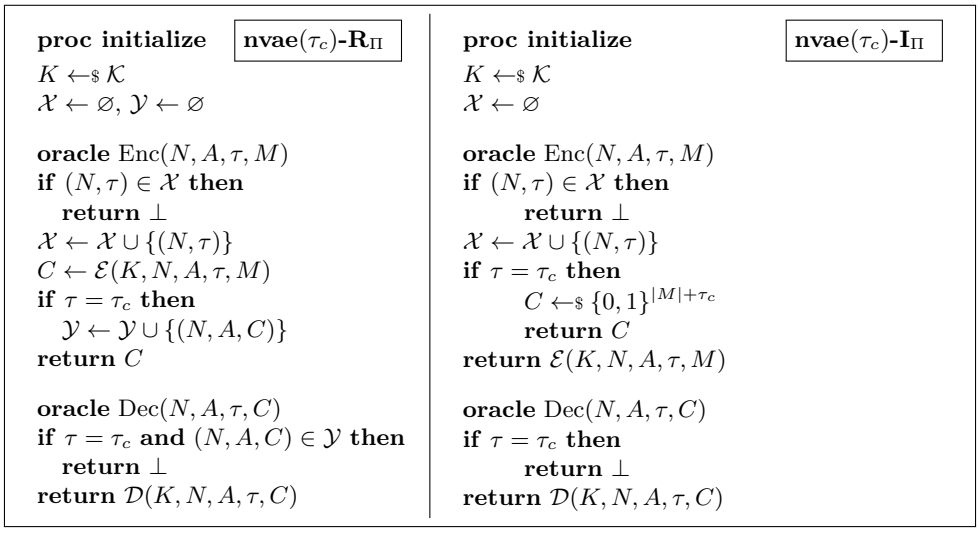


Figure 2: AE security with variable stretch. Security games for defining AE security of a nonce-based AE scheme $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ with variable-stretch.

$\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ through security games $\mathbf{nvae}(\tau_c)\text{-}\mathbf{R}_\Pi$ (left) and $\mathbf{nvae}(\tau_c)\text{-}\mathbf{I}_\Pi$ (right) in Figure 2, parameterized by the so-called challenge stretch value $\tau_c \in \mathcal{I}_T$, where the adversary $\mathscr{A}$ cannot repeat a nonce-stretch combination in its queries [RVV16]. As observed in Figure 2, a query, be it encryption or decryption, with a stretch other than τ_c will always be answered with the real encryption/decryption algorithm, giving the adversary extra information for distinguishing the real or ideal processing of queries stretched by τ_c. We refer the reader to the original publication for a detailed explanation of the **nvae** notion. We define the advantage of $\mathscr{A}$ in breaking the **nvae** security of Π with the target stretch τ_c as $\mathbf{Adv}_\Pi^{\mathbf{nvae}(\tau_c)}(\mathscr{A}) = \Pr[\mathscr{A}^{\mathbf{nvae}(\tau_c)\text{-}\mathbf{R}_\Pi} \Rightarrow 1] - \Pr[\mathscr{A}^{\mathbf{nvae}(\tau_c)\text{-}\mathbf{I}_\Pi} \Rightarrow 1]$.

We measure the resources of an **nvae** adversary in a fine-grained, vectorial fashion as $(t, \mathbf{q_e}, \mathbf{q_d}, \boldsymbol{\sigma}_\mathbf{e}, \boldsymbol{\sigma}_\mathbf{d})$, where t denotes the running time of the adversary, $\mathbf{q_e} = (q_e^\tau | \tau \in \mathcal{I}_T)$ denotes the vector that holds the number of encryption queries q_e^τ made with stretch τ for every stretch $\tau \in \mathcal{I}_T$, $\mathbf{q_d} = (q_d^\tau | \tau \in \mathcal{I}_T)$ denotes the same for the decryption queries, $\boldsymbol{\sigma}_\mathbf{e} = (\sigma_e^\tau | \tau \in \mathcal{I}_T)$ denotes the vector that holds the total amount of data σ^τ processed in all encryption queries with stretch τ for every $\tau \in \mathcal{I}_T$, while $\boldsymbol{\sigma}_\mathbf{d}$ denotes the same for the decryption queries. As indicated by Reyhanitabar et al., a typical analysis will use the resources related to τ_c (i.e. $q_e^{\tau_c}, q_d^{\tau_c}, \sigma_e^{\tau_c}, \sigma_d^{\tau_c}$) and aggregate resources q_e, q_d, q, σ_e, σ_d, σ with $q_e = \sum_{\tau \in \mathcal{I}_T} q_e^\tau$, $q_d = \sum_{\tau \in \mathcal{I}_T} q_d^\tau$, $q = q_e + q_d$, $\sigma_e = \sum_{\tau \in \mathcal{I}_T} \sigma_e^\tau$, $\sigma_d = \sum_{\tau \in \mathcal{I}_T} \sigma_d^\tau$ and $\sigma = \sigma_e + \sigma_d$. We additionally define the per-stretch aggregate variables $q^\tau = q_e^\tau + q_d^\tau$ and $\sigma^\tau = \sigma_e^\tau + \sigma_d^\tau$ for all $\tau \in \mathcal{I}_T$. Recall that an nae scheme Π with stretch τ is equivalent with an nvAE scheme with $\mathcal{I}_T = \{\tau\}$, so the vector-based adversarial resources become synonymous with the aggregate variables.

Finally, we informally call a scheme Π **nvae**-secure if for every $\tau_c \in \mathcal{I}_T$, for all "practical" values of resources $\mathbf{r}_{\tau_c}$ the advantage $\mathbf{Adv}_\Pi^{\mathbf{nvae}(\tau_c)}(\mathbf{r}_{\tau_c})$ is "small".

Key-equivalent separation by stretch. Reyhanitabar et al. proposed a notion that captures the intuition that changing the value of stretch with a single key is equivalent to having an independent key per stretch. The notion was primarily proposed to facilitate **nvae** security proofs for schemes known to be **nae** secure. The **kess** property of an nvAE scheme $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ is defined through indistinguishability of the games defined Figure 3 by an adversary $\mathscr{A}$. The advantage of $\mathscr{A}$ is defined as

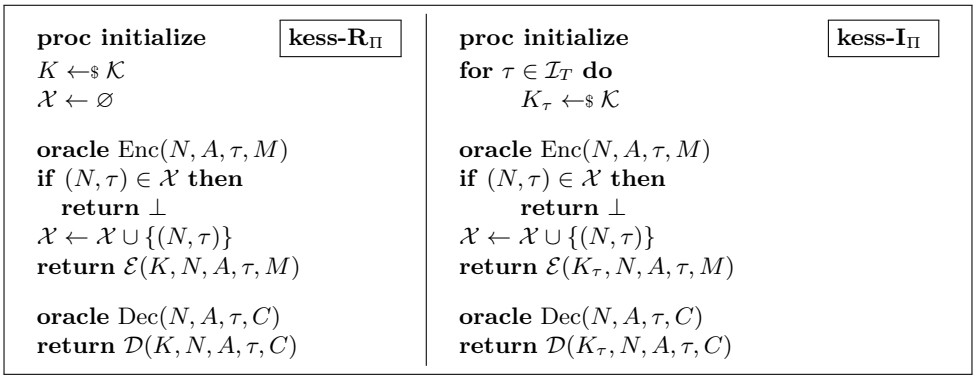


Figure 3: Key-equivalent separation by stretch. Games defining **kess** property of a nonce-based AE scheme $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ with variable stretch. Note that the independent keying for each $\tau \in \mathcal{I}_T$ in game **kess-I**$_\Pi$ can be done by lazy sampling if needed.

$\mathbf{Adv}^{\mathbf{kess}}_{\Pi}(\mathscr{A}) = \Pr\left[\mathscr{A}^{\mathbf{kess\text{-}R}_\Pi} \Rightarrow 1\right] - \Pr\left[\mathscr{A}^{\mathbf{kess\text{-}I}_\Pi} \Rightarrow 1\right]$. The adversarial resources of interest for the **kess** notion are $(t, \mathbf{q_e}, \mathbf{q_d}, \boldsymbol{\sigma})$, as defined for the **nvae**(τ_c) notion in the current Section. Reyhanitabar et al. proved Theorem 1; to prove that an **nae**-secure AE scheme is **nvae**-secure, it suffices to show that the **kess** advantage of any adversary is small.

Theorem 1 (**kess** $\wedge$ **nae** $\Rightarrow$ **nvae**). *Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be a nonce-based AE scheme with variable stretch. We have that*

$$\mathbf{Adv}^{\boldsymbol{nvae}(\tau_c)}_{\Pi}(t, \mathbf{q_e}, \mathbf{q_d}, \boldsymbol{\sigma}) \leq \mathbf{Adv}^{\boldsymbol{kess}}_{\Pi}(t', \mathbf{q_e}, \mathbf{q_d}, \boldsymbol{\sigma}) + \mathbf{Adv}^{\boldsymbol{nae}}_{\Pi[\tau_c]}(t'', q_e^{\tau_c}, q_d^{\tau_c}, \sigma^{\tau_c}),$$

with $t' = t + O(q)$ and $t'' = t + O(\sigma)$ where $q = \sum_{\tau \in \mathcal{I}_T}(q_e^\tau + q_d^\tau)$ and $\sigma = \sum_{\tau \in \mathcal{I}_T}(\sigma_e^\tau + \sigma_d^\tau)$.

3 Counter with CBC-MAC(CCM)

CTR + CBC-MAC is a nonce-based AE scheme proposed by Whiting,Housley, and Ferguson [DWF03]. It is parameterized by a blockcipher $\mathbf{E} : \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ with $n = 128$,[1] a tag length $\tau \in \{32, 48, 64, 80, 96, 112, 128\}$, and a nonce length $\nu \in \{56, 64, 72, 80, 88, 96, 104\}$. A trade-off must be made between the nonce length ν and the maximal message length $8 \cdot (2^{120-\nu} - 1)$ bits. The AD and message space of $\mathbf{CCM}[\mathbf{E}, \tau, \nu]$ are respectively $\mathcal{A} = \bigcup_{i=0}^{2^{64}-1} \mathsf{B}^i$ and $\mathcal{M} = \bigcup_{i=0}^{2^{120-\nu}-1} \mathsf{B}^i$. The encryption and the decryption algorithms of $\mathbf{CCM}[\mathbf{E}, \tau, \nu]$ are described in Figure 4.[2] Internally, CCM can be split in two parts.

Tag Generation with CBC-MAC. The tag generation step (upper part of Figure 5) uses plain CBC-MAC to generate a 128-bit tag, which is then truncated to the desired tag length τ. The string processed by the plain CBC-MAC is constructed as $B = \mathsf{encB}_0(N, A, M) \| \mathsf{encA}(A) \| \mathsf{encM}(M)$, with

$$\mathsf{encB}_0(N, A, M) = \mathsf{flags}(A) \| N \| \langle |M|_8 \rangle_{120-\nu} \text{ (outputs a single block)},$$
$$\mathsf{flags}(A) = 0 \| \mathbb{1}_{A \neq \varepsilon} \| \langle \tau/16 - 1 \rangle_3 \| \langle 14 - \nu/8 \rangle_3 \text{ (outputs a single byte)},$$

[1] CCM is only defined to for use with 128-bit blockciphers [DWF03].

[2] Note that we deviate from he notation in the riginal publication [DWF03] and the CCM RFC [WHF03], as the descriptions in these publications are incompatible with the common notation of AEAD.

Algorithm $\mathcal{E}_K(N, A, M)$
$B_0 \leftarrow \mathsf{encB}_0(N, A, M)$
$B_1||\cdots||B_\ell \leftarrow \mathsf{encA}(A)\|\mathsf{encM}(M)$
$Y_0 \leftarrow \mathrm{E}_K(B_0)$
for $i \leftarrow 1$ **to** ℓ **do**
  $Y_i \leftarrow \mathrm{E}_K(B_i \oplus Y_{i-1})$
$T = \mathsf{left}_\tau(Y_\ell)$
$M_1\|M_2\cdots\|M_m \leftarrow M$ where
  $|M_m| \le n$ and $|M_i| = n$ otherwise
for $i \leftarrow 1$ **to** m **do**
  $Z_i \leftarrow \mathsf{ctr}(N, i)$
  $C_i \leftarrow \mathsf{left}_{|M_i|}(\mathrm{E}_K(Z_i)) \oplus M_i$
$C \leftarrow C_1\|C_2\|\cdots\|C_m$
$Z_0 \leftarrow \mathsf{ctr}(N, 0)$
$C \leftarrow C\|\mathsf{left}_\tau(\mathrm{E}_K(Z_0)) \oplus T$
return C

Algorithm $\mathcal{D}_K(N, A, C)$
$C_1\|C_2\cdots C_m\|T \leftarrow C$ where
  $|C_m| \le n$ and $|C_i| = n$ otherwise
  and where $|T| = \tau$
for $i \leftarrow 1$ **to** m **do**
  $Z_i \leftarrow \mathsf{ctr}(N, i)$
  $M_i \leftarrow \mathsf{left}_{|C_i|}(\mathrm{E}_K(Z_i)) \oplus C_i$
$M = M_1||M_2||\cdots||M_m$
$B_0 \leftarrow \mathsf{encB}_0(N, A, M)$
$B_1||\cdots||B_\ell \leftarrow \mathsf{encA}(A)\|\mathsf{encM}(M)$
$Y_0 \leftarrow \mathrm{E}_K(B_0)$
for $i \leftarrow 1$ **to** ℓ **do**
  $Y_i \leftarrow \mathrm{E}_K(B_i \oplus Y_{i-1})$
$Z_0 \leftarrow \mathsf{ctr}(N, 0)$
$T' = \mathsf{left}_\tau(Y_\ell \oplus \mathrm{E}_K(Z_0))$
if $T = T'$ **then**
  return M
return $\perp$

Figure 4: CCM$[E, \tau, \nu]$ mode for AEAD, with $E : \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ a blockcipher. The functions $\mathsf{encB}_0()$, $\mathsf{encA}()$,$\mathsf{encM}()$ and $\mathsf{ctr}()$ are defined in Section 3.

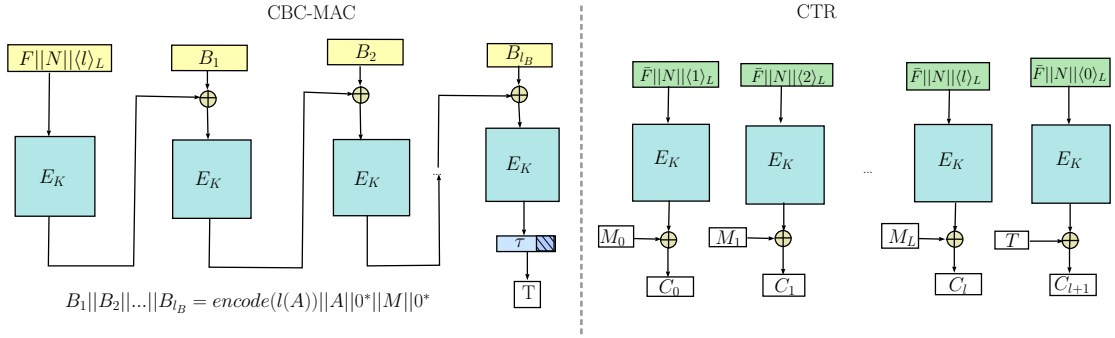


Figure 5: Illustration of the inner workings of CCM. The top half depicts the computation of the tag using CBC-MAC and the bottom half depicts the encryption process using the CTR mode. enc(A) is a prefix free encoding of A, while r is an integer such that $|enc(M)| + r$ is divisible by n

$\mathbb{1}_{A\neq\varepsilon} = 1$ iff $A \neq \varepsilon$ and $\mathbb{1}_{A\neq\varepsilon} = 0$ otherwise, $\mathsf{encA} : \mathsf{B}^* \to (\{0,1\}^n)^*$ a prefix-free encoding function that extends the length by at most a single block (i.e., $|\mathsf{encA}(A)|/n \le \lceil |A|/n \rceil + 1$), and $\mathsf{encM}(M) = M \,\|\, 0^r$ for the integer $0 \le r < n$ that makes $|\mathsf{encM}(M)|$ divisible by n. We note that the composed encoding function itself $\mathsf{encB}_0(N, A, M)\|\mathsf{encA}(A)\|\mathsf{encM}(M)$ is prefix-free [Jon02]. The output of the plain CBC-MAC is then masked with a key stream block computed in the second, CTR-mode part.

CTR Mode Encryption. The ciphertext blocks are computed with CTR mode (lower part of Figure 5). The i^{th} input counter block, used to generate the i^{th} key stream block, is computed as $\mathsf{ctr}(N, i) = \mathsf{flags}'\|N\|\langle i\rangle_{120-\nu}$, where $\mathsf{flags}' = 0^5\|\langle 14 - \nu/8\rangle_3$ is a one-byte string.

The security of the standard **CCM** scheme was proven by Jonsson in [Jon02] in the following theorem:

Theorem 2 ([Jon02])**.** *For a blockcipher $E : \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ and any valid values*

```
101: Algorithm  E_K(N, A, τ, M)
102:     N' ← N||⟨τ/8⟩_8
103:     C = CCM[E, τ, ν + 8].E_K(N', A, M)
104:     return C

201: Algorithm  D_K(N, A, τ, C)
202:     N' ← N||⟨τ/8⟩_8
203:     M = CCM[E, τ, ν + 8].D_K(N', A, C)
204:     return M
```

Figure 6: vCCM$[E, \mathcal{I}_T, \nu]$ mode for AEAD, with $E : \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ a blockcipher.

of τ and ν we have

$$\mathbf{Adv}^{auth}_{\boldsymbol{CCM}[E,\tau,\nu]}(t, q_e, q_d, \sigma_e, \sigma_d) \le \mathbf{Adv}^{\boldsymbol{prf}}_{\mathbf{E}}(t', \sigma') + \frac{q_d}{2^\tau} + \frac{(2\sigma + 3q)^2}{2^{n+1}}$$

$$\mathbf{Adv}^{priv}_{\boldsymbol{CCM}[E,\tau,\nu]}(t, q_e, \sigma_e) \le \mathbf{Adv}^{\boldsymbol{prf}}_{\mathbf{E}}(t', \sigma'') + \frac{(2\sigma_e + 3q_e)^2}{2^{n+1}}$$

where $\sigma' \le 2\sigma + 3q$, $\sigma'' \le 2\sigma_e + 3q_e$ and $t' \le t + \gamma \cdot \sigma$ for some "small" constant γ.

There is a seeming inflation of the bound compared to Jonsson's theorems. This is due to the difference in resource counting: while Jonsson defines the equivalents of σ and σ_e already including the overhead due to CCM's structure, we prefer to treat σ as data complexity as perceived by the user, who doesn't need to understand the internals of CCM. Combining the bounds of Theorem 2 yields the following **nae** security bound.

Corollary 1 ([RS06])**.** *For a blockcipher $E : \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ and any valid values of τ and ν we have*

$$\mathbf{Adv}^{\boldsymbol{nae}}_{\boldsymbol{CCM}[E,\tau,\nu]}(t, q_e, q_d, \sigma_e, \sigma_d) \le \mathbf{Adv}^{\boldsymbol{prf}}_{\mathbf{E}}(t', \sigma') + \frac{q_d}{2^\tau} + \frac{(2\sigma + 3q)^2}{2^{n+1}} + \frac{(2\sigma_e + 3q_e)^2}{2^{n+1}}$$

where $\sigma' \le 2\sigma + 3q$ and $t' \le t + \gamma \cdot \sigma$ for some "small" constant γ.

4 Variable Tag CCM (vCCM)

vCCM is a nonce-based AE scheme parameterized by a blockcipher $\mathbf{E} : \mathcal{K}_\tau \times \{0,1\}^n \to \{0,1\}^n$ with $n = 128$, an ordered set of non-zero tag lengths $\mathcal{I}_T \subseteq \{32, 48, 64, 80, 96, 112, 128\}$, and a nonce length $\nu \in \{56, 64, 72, 80, 88, 96\}$. There is a trade-off between the nonce length ν and the maximal message length of $8 \cdot (2^{(112-\nu)} - 1)$ bits. The associated data and message space of **vCCM**$[\mathbf{E}, \mathcal{I}_T, \nu]$ are respectively $\mathcal{A} = \bigcup_{i=0}^{2^{64}-1} \mathsf{B}^i$ and $\mathcal{M} = \bigcup_{i=0}^{2^{112-\nu}-1} \mathsf{B}^i$. The encryption and the decryption algorithms of **vCCM**$[\mathbf{E}, \mathcal{I}_T, \nu]$ are described in Figure 6 and illustrated in Figure 7.

.

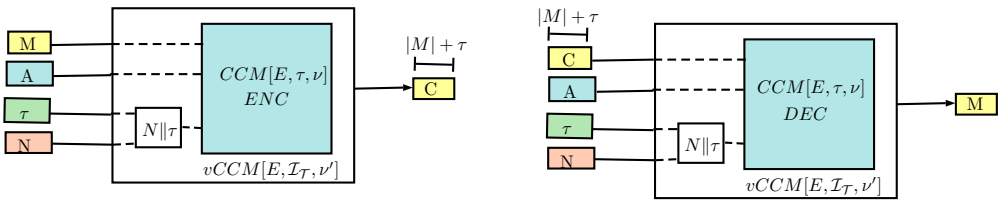


Figure 7: Illustration of the black box transform of **CCM** to **vCCM**. Here, $\nu' = \nu - 8$.

The Transform. The nvAE scheme **vCCM** is obtained as a *black box transform* of

CCM, requiring no modifications of the standard **CCM**. This property has been our primary design goal, as it is key for the ability of the new nvAE scheme to benefit from existing software and hardware implementations, and thus be instantly used at a massive scale.

The options available for a black-box transform are, informally speaking, injecting the tag length into the nonce, the associated data, or the key (not the message, due to undesirable additional ciphertext expansion). With the option of tag-dependent key undesirable, as discussed in the publication introducing the **nvae** model [RVV16], one is left with a tag-dependent nonce, tag-dependent associated data, or a combination thereof.

In the particular case of CCM, injecting the stretch in the associated data does not work; when the same message was encrypted with different tag lengths, the change would propagate through the computation of CBC-MAC, and hence only to the last block of the ciphertext. A trivial distinguishing attack could be mounted using the non-tag ciphertext blocks.

On the other hand, injecting the stretch into the nonce propagates to every block of ciphertext, thanks to the structure of CCM, as nonce is encoded into the initial bock B_0 processed by the plain CBC-MAC, as well as into each counter block, used as blockcipher input to compute the key stream blocks. This being an efficient and security-wise sufficient option, our transform consists in simply dedicating the last byte of the **CCM** nonce to containing an encoding of the used tag length.

5 Security of vCCM

In this section, we formally state and prove the nvae security of **vCCM** in Theorem 3. The analysis is based on Theorem 1, allowing to reuse the result on nae security of CCM by Jonsson [Jon02] and leaving only the kess property of **vCCM** to be analyzed. We formally state and prove the latter in Lemma 1 using the technique of code-based games [BR06].

Theorem 3. *Let* $\mathrm{E} : \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ *with* $n = 128$ *be a blockcipher,* $\mathcal{I}_T \subseteq \{32, 48, 64, 80, 96, 112, 128\}$ *and* $\nu \in \{56, 64, 72, 80, 88, 96\}$. *Then the following inequality holds:*

$$\begin{aligned}\mathbf{Adv}^{\boldsymbol{nvae}[\tau_c]}_{\boldsymbol{vCCM}[E,\mathcal{I}_T,\nu]}(t, \mathbf{q_e}, \mathbf{q_d}, \boldsymbol{\sigma}_\mathbf{e}, \boldsymbol{\sigma}_\mathbf{d}) \leq & |\mathcal{I}_T| \cdot \mathbf{Adv}^{\boldsymbol{prp}}_{\mathrm{E}}(t', \sigma') + 3 \cdot \frac{(2\sigma + 3q)^2}{2^n} \\ & + \frac{q_d}{2^\tau} + \frac{(2\sigma^{\tau_c} + 3q^{\tau_c})^2}{2^{n+1}} + \frac{(2\sigma_e^{\tau_c} + 3q_e^{\tau_c})^2}{2^{n+1}}\end{aligned}$$

where $\sigma' \leq 2\sigma + 3q$ *and* $t' \leq t + \sigma' \cdot \gamma$ *for a "small" constant* γ, *and the adversarial resources are as defined in Section 2.*

Corollary 2. *As* $|\mathcal{I}_T| \leq 7$, *we directly have*

$$\begin{aligned}\mathbf{Adv}^{\boldsymbol{nvae}[\tau_c]}_{\boldsymbol{vCCM}[E,\mathcal{I}_T,\nu]}(t, \mathbf{q_e}, \mathbf{q_d}, \boldsymbol{\sigma}_\mathbf{e}, \boldsymbol{\sigma}_\mathbf{d}) \leq & 7 \cdot \mathbf{Adv}^{\boldsymbol{prp}}_{\mathrm{E}}(t', \sigma') + 3 \cdot \frac{(2\sigma + 3q)^2}{2^n} \\ & + \frac{q_d}{2^\tau} + \frac{(2\sigma^{\tau_c} + 3q^{\tau_c})^2}{2^{n+1}} + \frac{(2\sigma_e^{\tau_c} + 3q_e^{\tau_c})^2}{2^{n+1}}\end{aligned}$$

Proof. We start by replacing the block cipher in both the $\mathbf{nvae}(\tau_c)$**-R** and the $\mathbf{nvae}(\tau_c)$**-I** game by a secret random permutation $\pi \leftarrow\!\!\$\ \mathrm{Perm}(n)$ to obtain the following inequality

$$\mathbf{Adv}^{\mathbf{nvae}(\tau_c)}_{\mathbf{vCCM}[E,\mathcal{I}_T,\nu]}(t, \mathbf{q_e}, \mathbf{q_d}, \boldsymbol{\sigma}_\mathbf{e}, \boldsymbol{\sigma}_\mathbf{d}) \leq \mathbf{Adv}^{\mathbf{nvae}(\tau_c)}_{\mathbf{vCCM}[\pi,\mathcal{I}_T,\nu]}(t, \mathbf{q_e}, \mathbf{q_d}, \boldsymbol{\sigma}) + |\mathcal{I}_T| \cdot \mathbf{Adv}^{\mathbf{prp}}_{E}(t', \sigma')\,,$$

with $\sigma' \leq 2\sigma + 3q$ and $t' \leq t + \sigma' \cdot \gamma$, as there is a single instance of E used in the game $\mathbf{nvae}[\tau_c]$**-R** and $|\mathcal{I}_T| - 1$ instances of E (used with independent keys) in the game $\mathbf{nvae}[\tau_c]$**-I**, each of them processing no more than $2\sigma + 3q$ blocks in total. This is given by the fact that a CCM encryption (resp. decryption query) with a AD blocks and m

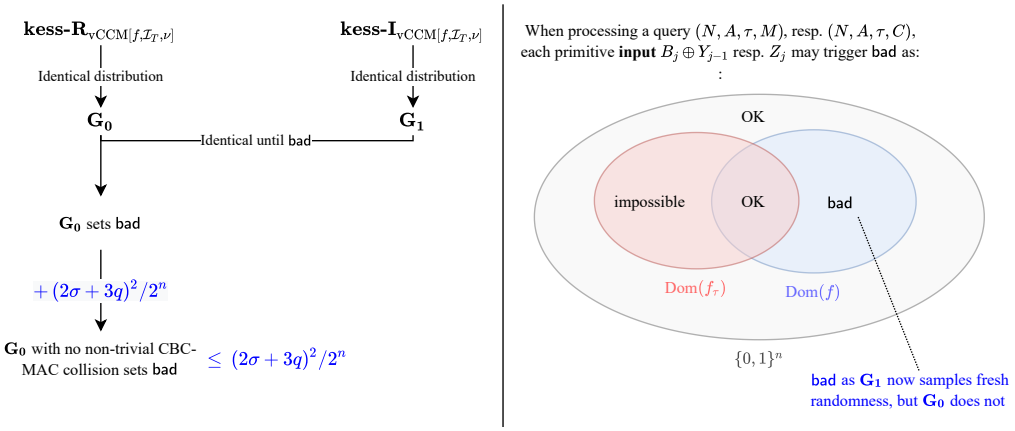


Figure 8: Outline of the proof of Lemma 1 and the visualization of the main bad event.

message blocks requires $a + 2m + 3$ blockcipher calls in the worst case, two extra calls being necessary for processing of B_0 and computing the keystream block for the tag, and the third extra call being possibly induced by the length increase of $\mathsf{encA}(\cdot)$. Note that in the $\mathbf{nvae}[\tau_c]$**-I**, there are now $|\mathcal{I}_T| - 1$ *independent* permutations.

Using the standard RP-RF switching lemma [BR06], the random permutation π is further replaced in both $\mathbf{nvae}[\tau_c]$ games by a secret random function $f \leftarrow\!\$\ \mathrm{Func}(n)$ to obtain

$$\mathbf{Adv}^{\mathbf{nvae}(\tau_c)}_{\mathbf{vCCM}[\pi,\mathcal{I}_T,\nu]}(t, \mathbf{q_e}, \mathbf{q_d}, \boldsymbol{\sigma}_\mathbf{e}, \boldsymbol{\sigma}_\mathbf{d}) \leq \mathbf{Adv}^{\mathbf{nvae}(\tau_c)}_{\mathbf{vCCM}[f,\mathcal{I}_T,\nu]}(t, \mathbf{q_e}, \mathbf{q_d}, \boldsymbol{\sigma}) + 2 \cdot \frac{(2\sigma+3q)^2}{2^{n+1}}.$$

This is because the secret permutation π in $\mathbf{nvae}[\tau_c]$**-R** is used no more than $(2\sigma + 3q)$ times, contributing $\frac{(2\sigma+3q)^2}{2^{n+1}}$ to the bound, and the $|\mathcal{I}_T| - 1$ permutations in $\mathbf{nvae}[\tau_c]$**-I** are used on no more than $(2\sigma + 3q)$ blocks *in total*, so the sum of their individual switching contributions is upper bounded by $\frac{(2\sigma+3q)^2}{2^{n+1}}$ as well.
Next, we use Theorem 1 to get the following inequality

$$\begin{aligned}\mathbf{Adv}^{\mathbf{nvae}(\tau_c)}_{\mathbf{vCCM}[f,\mathcal{I}_T,\nu]}(\mathbf{q_e}, \mathbf{q_d}, \boldsymbol{\sigma}_\mathbf{e}, \boldsymbol{\sigma}_\mathbf{d}) \leq\ & \mathbf{Adv}^{\mathbf{kess}}_{\mathbf{vCCM}[f,\mathcal{I}_T,\nu]}(\mathbf{q_e}, \mathbf{q_d}, \boldsymbol{\sigma}_\mathbf{e}, \boldsymbol{\sigma}_\mathbf{d}) \\ & + \mathbf{Adv}^{\mathbf{nae}}_{\mathbf{vCCM}[f,\{\tau_c\},\nu]}(q_e^{\tau_c}, q_d^{\tau_c}, \sigma_e^{\tau_c}, \sigma_d^{\tau_c})\ .\end{aligned}$$

From Corollary 1 and the fact that $\mathbf{Adv}^{\mathbf{prf}}_f(\mathscr{A}) = 0$, the following inequality holds:

$$\mathbf{Adv}^{\mathbf{nae}}_{\mathbf{vCCM}[f,\{\tau_c\},\nu]}(q_e^{\tau_c}, q_d^{\tau_c}, \sigma_e^{\tau_c}, \sigma_d^{\tau_c}) \leq \frac{q_d}{2^\tau} + \frac{(2\sigma^{\tau_c} + 3q^{\tau_c})^2}{2^{n+1}} + \frac{(2\sigma_e^{\tau_c} + 3q_e^{\tau_c})^2}{2^{n+1}}$$

The proof is finalized by applying Lemma 1 to upper bound $\mathbf{Adv}^{\mathbf{kess}}_{\mathbf{vCCM}[f,\mathcal{I}_T,\nu]}(t, \mathbf{q_e}, \mathbf{q_d}, \boldsymbol{\sigma}_\mathbf{e}, \boldsymbol{\sigma}_\mathbf{d})$. □

By using the **kess**−security definition in Figure 3 and the games in Figure 9, we obtain the following bound.

Lemma 1. *Let $f : \{0,1\}^n \to \{0,1\}^n$ be a random function $f \leftarrow\!\$\ \mathrm{Func}(n)$ with $n = 128$, $\mathcal{I}_T \subseteq \{32, 48, 64, 80, 96, 112, 128\}$ and $\nu \in \{56, 64, 72, 80, 88, 96\}$. Then the following inequality holds:*

$$\mathbf{Adv}^{\mathit{kess}}_{\mathit{vCCM}[f,\mathcal{I}_T,\nu]}(\mathbf{q_e}, \mathbf{q_d}, \boldsymbol{\sigma}_\mathbf{e}, \boldsymbol{\sigma}_\mathbf{d}) \leq 2 \cdot (2\sigma + 3q)^2/2^n\ .$$

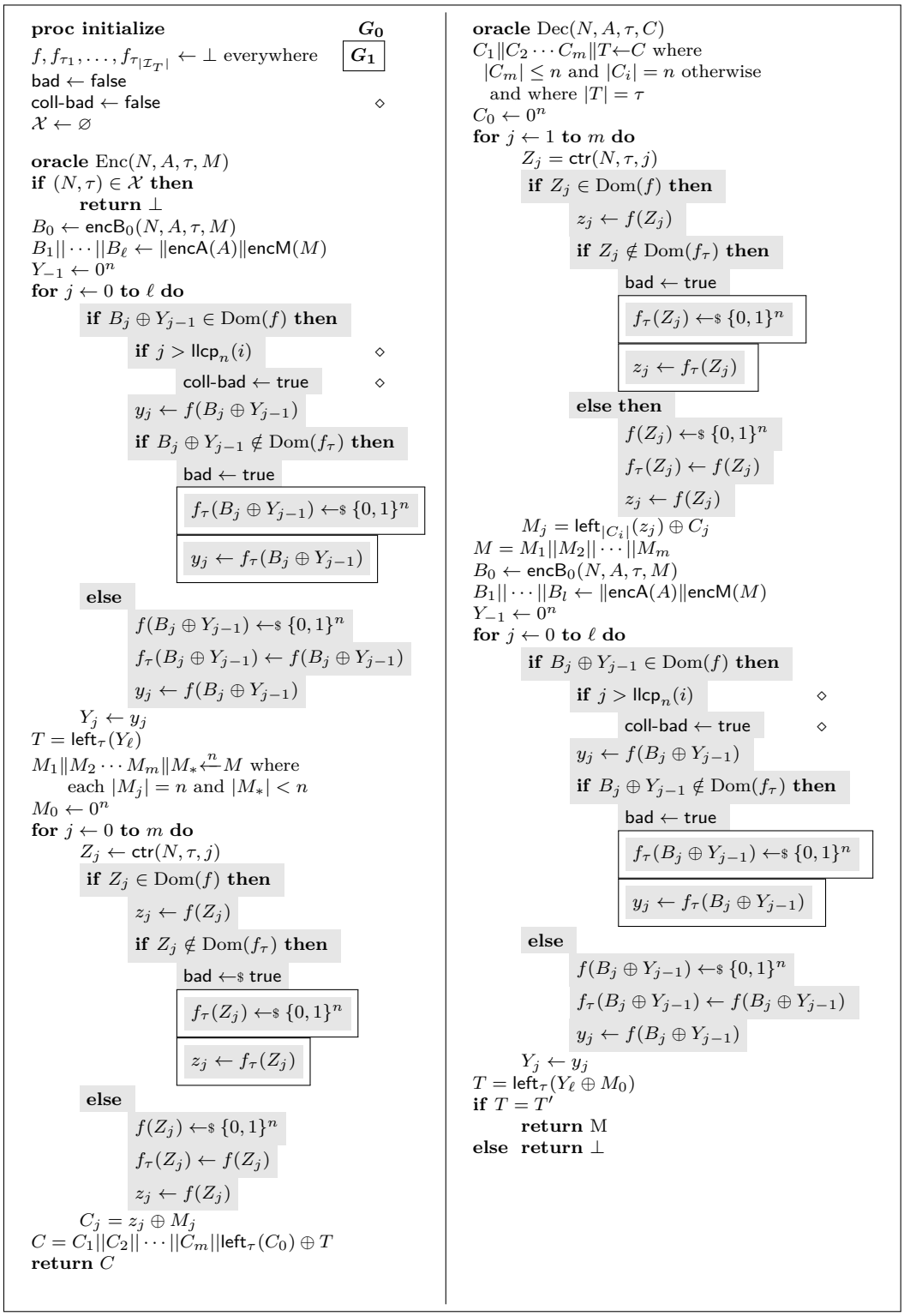


Figure 9: Games Definition Adversarial games $\boldsymbol{G_0}$ and $\boldsymbol{G_1}$ for the proof of Lemma 1. $\boldsymbol{G_0}$ is obtained by omitting the boxed statements and the statements marked by $\diamond$, while $\boldsymbol{G_1}$ is obtained by including the boxed statements and omitting the statements marked by $\diamond$. Here, we let for a partially defined function f, $\mathrm{Dom}(f)$ denote the set $\{x \in \{0,1\}^n | f(x) \neq \perp\}$. The functions $\mathsf{encB}_0()$ and $\mathsf{ctr}()$ are extended to process τ in the obvious way. For definition of $\mathsf{llcp}_n(i)$ see proof of Lemma 1.

Proof. The proof is based on the games $\boldsymbol{G_0}$ and $\boldsymbol{G_1}$ in Figure 9. The code in Figure 9 is obtained by using **vCCM** to instantiate the Enc and Dec oracles of the **kess** games (Figure 2), such that in the place of each invocation of the underlying primitive, there is a nested if-else block (lines with gray background). In the following we will refer to the underlying primitive used to instantiate **vCCM** simply by "primitive". The proof outline is visualized in Figure 8.

Simulating the kess games. Roughly speaking, these if-else blocks ensure that the supposed primitive outputs are being sampled such that they are simultaneously compatible with a single random function f and $|\mathcal{I}_T|$ independent random functions $f_{\tau_1}, \ldots, f_{\tau_{|\mathcal{I}_T|}}$, for as long as possible. When it is no longer possible to maintain this double compatibility, both games set flag bad to true. Formally, we claim that the games $\boldsymbol{G_0}$ and $\boldsymbol{G_1}$ are in fact implementing $\textbf{kess-R}_{\textbf{vCCM}[f,\mathcal{I}_T,\nu]}$ and $\textbf{kess-I}_{\textbf{vCCM}[f,\mathcal{I}_T,\nu]}$, respectively.

This is easy to see for $\boldsymbol{G_0}$. When we omit the boxed statements, then in each query, the primitive-output variable y_j, respectively z_j, is assigned a previously sampled value whenever the input to the primitive ($B_j \oplus Y_{j-1}$, resp. Z_j) is already in $\mathrm{Dom}(f)$, and freshly sampled and used to extend f otherwise.

For $\boldsymbol{G_1}$, the boxed statements are included, ensuring that in each query, in the special case when the input to the primitive ($B_j \oplus Y_{j-1}$, resp. Z_j) is in $\mathrm{Dom}(f)$ but not in $\mathrm{Dom}(f_\tau)$, the value of the primitive-output variable y_j, respectively z_j is freshly re-sampled, with τ being the amount of stretch in current query. We note that the seemingly missing case when an input to the primitive ($B_j \oplus Y_{j-1}$, resp. Z_j) is present in $\mathrm{Dom}(f_\tau)$ but is missing in $\mathrm{Dom}(F)$ (or else $\mathrm{Dom}(f_\tau)\backslash\mathrm{Dom}(f) \neq \emptyset$) cannot occur. This is shown by a simple induction: at initialization, all functions are undefined everywhere, and so $\mathrm{Dom}(f_\tau)\backslash\mathrm{Dom}(f) = \emptyset$. Every time the primitive output is computed in $\boldsymbol{G_1}$ in a query with stretch τ, if f_τ is extended by a preimage-image pair $x, f_\tau(x)$, then $f(x)$ is defined at the same time, or $f(x)$ has already been defined before.

We thus have for any adversary $\mathscr{A}$ that $\mathbf{Adv}^{\mathbf{kess}}_{\mathbf{vCCM}[f,\mathcal{I}_T,\nu]}(\mathscr{A}) = \Pr[\mathscr{A}^{\boldsymbol{G_0}} \Rightarrow 1] - \Pr[\mathscr{A}^{\boldsymbol{G_1}} \Rightarrow 1]$. Further, because the games $\boldsymbol{G_0}$ and $\boldsymbol{G_1}$ are identical until the flag bad is set to true, we have by the fundamental lemma of game-playing [BR06]

$$\Pr[\mathscr{A}^{\boldsymbol{G_0}} \Rightarrow 1] - \Pr[\mathscr{A}^{\boldsymbol{G_1}} \Rightarrow 1] \leq \Pr[\mathscr{A}^{\boldsymbol{G_0}} \text{ sets } \mathsf{bad}] \ ,$$

where the event "sets bad" is defined as the flag bad being true at the end of the experiment. When flag bad is set, the game $\boldsymbol{G_1}$ diverges from $\boldsymbol{G_0}$, as visualized in Figure 8: the former samples a fresh primitive output, while the latter reuses a previously sampled one, and their distributions are no longer equivalent.

Characterizing the experiment. We start by characterizing the interaction of an adversary $\mathscr{A}$ with its oracles as defined in Figure 9. We let N^i, A^i, τ_i, M^i denote C^i denote the nonce, the AD, the stretch, the (internally computed) plaintext and the ciphertext of the i^{th} query. We denote by $B^i_0, \ldots, B^i_{\ell_i}$ the input blocks fed to plain CBC-MAC in the i^{th} query, by $Y^i_1, \ldots, Y^i_{\ell_i}$ the corresponding outputs, such that $Y^i_j = y^i_j$ for $1 \leq j \leq \ell_i$ and $T^i = Y^i_{\ell_i}$. $Y^i_{-1} = 0^n$ is a special all-zero block in every query. We further let $M^i_1, \ldots, M^i_{m_i}$ denote the message blocks, $C^i_1, \ldots, C^i_{m_i}$ the corresponding ciphertext blocks, $Z^i_0, \ldots Z^i_{m_i}$ the counter blocks, and $z^i_0, \ldots, z^i_{m_i}$ the corresponding primitive outputs, such that we have $C^i_j = M^i_j \oplus z^i_j$ for $1 \leq j \leq m_i$ and $C^i_0 \oplus T^i = z^i_0 \oplus T^i$. This notation is valid irrespective of whether the i^{th} query is an encryption or a decryption query.

We assume that the adversary $\mathscr{A}$ makes no pointless queries; i.e., $\mathscr{A}$ makes no repeated queries, does not make a decryption query N, A, C after a previous encryption query N, A, M returned the ciphertext C, and does not make an encryption query N, A, M after a previous decryption query N, A, C returned $M \neq \perp$. This is without generality, as the responses to these queries are all trivially known.

CBC collisions. To bound the probability $\Pr[\mathscr{A}^{\boldsymbol{G_0}} \text{ sets } \mathsf{bad}]$, we first define an auxiliary flag $\mathsf{coll\text{-}bad}$, and extend the games $\boldsymbol{G_0}$ and $\boldsymbol{G_1}$ with the lines marked by the symbol $\diamond$ in Figure 9. Here, the notation $\mathsf{llcp}_n(i)$ is a shorthand for $\mathsf{llcp}_n(B^1, \ldots, B^{i-1}; B^i) - 1$, the index of the first block of the input to the plain CBC-MAC in the i^{th} query $B^i = B^i_0 \| \ldots \| B^i_\ell$ that is beyond the longest common blockwise prefix with any of such inputs in the previous $i-1$ queries. Note that one is subtracted here because the blocks of B^i are indexed from zero.

Since $\mathscr{A}$ makes no trivial queries and because the string B^i is a prefix-free encoding of (N, A, M) in CCM, and consequently of (N, A, τ, M) in **vCCM**, we have $\mathsf{llcp}_n(i) < \ell_i$ for every $1 \le i \le q$. I.e., each B^i is distinct from all prefixes of $B^1, \ldots, B^{i-1}$. In particular, the value of $\mathsf{llcp}_n(i)$ is independent of the game's randomness; when the i^{th} query is a decryption query, the value of $\mathsf{llcp}_n(i)$ can be computed using the ciphertexts stripped of the tag as substitutes for the plaintext as $\mathsf{llcp}_n(\bar{B}^1, \ldots, \bar{B}^{i-1}; \bar{B}^i)$ where $\bar{B}^r = \mathsf{encB}_0(N^r, A^r, \tau_r, \mathsf{left}_{|C^r|-\tau}(C^r)) \| \mathsf{encA}(A) \| \mathsf{encM}(\mathsf{left}_{|C^r|-\tau}(C^r)) - 1$ for $1 \le r \le i$. This works, because $\mathsf{encB}_0(N^r, A^r, \tau_r, \mathsf{left}_{|C^r|-\tau}(C^r))$ only depends on the length of $\mathsf{left}_{|C^r|-\tau}(C^r)$, and because the xor-based CTR encryption preserves the value of $\mathsf{llcp}_n(\mathsf{encM}(\mathsf{left}_{|C^r|-\tau}(C^r)), \mathsf{encM}(\mathsf{left}_{|C^i|-\tau}(C^i))) = \mathsf{llcp}_n(\mathsf{encM}(M^r), \mathsf{encM}(M^i))$ if $(N^r, \tau_r, |C^r|) = (N^i, \tau_i, |C^i|)$. If the last condition is not met, the plaintext, resp. ciphertext blocks are irrelevant for the value of $\mathsf{llcp}_n(i)$.

In a nutshell, the $\mathsf{coll\text{-}bad}$ corresponds to reusing an input $B^i_j \oplus y^i_{j-1}$ to the function f (i.e., evaluating f on a point already in $\mathrm{Dom}(f)$) beyond what is trivially determined by the common prefix with previous queries. We observe that

$$\Pr[\mathscr{A}^{\boldsymbol{G_0}} \text{ sets } \mathsf{bad}] \le \Pr[\mathscr{A}^{\boldsymbol{G_0}} \text{ sets } \mathsf{coll\text{-}bad}] + \Pr[\mathscr{A}^{\boldsymbol{G_0}} \text{ sets } \mathsf{bad} | \neg \mathscr{A}^{\boldsymbol{G_0}} \text{ sets } \mathsf{coll\text{-}bad}]$$

and proceed to bound $\Pr[\mathscr{A}^{\boldsymbol{G_0}} \text{ sets } \mathsf{coll\text{-}bad}]$ by induction on blocks y^i_j, with the induction assumption being that $\mathsf{coll\text{-}bad} = \mathsf{false}$ when the block is about to be processed. When i^{th} query is being processed, $\mathsf{coll\text{-}bad}$ can only be set when sampling the values of y^i_j for $\mathsf{llcp}_n(i) + 1 \le j \le \ell_i$. For each such value of j, we bound the probability of collision between $B^i_j \oplus Y^i_{j-1}$ and each element already in $\mathrm{Dom}(f)$, with help of the following case analysis.

We have three base cases for the value of j, namely $j = 0$, $0 < j = \mathsf{llcp}_n(i) + 1$ and $\mathsf{llcp}_n(i) + 1 < j \le \ell_i$. Before examining them, we note that the domain points of f, $x \in \mathrm{Dom}(f)$, can be classified in three "types". We say x is of $\mathsf{type1}$ when it has been added as the initial CBC-MAC input block $B^{i'}_0 = \mathsf{encB}_0(N^{i'}, A^{i'}, \tau_{i'}, M^{i'})$ for some $i' < i$. We say x is of $\mathsf{type2}$ when it has been added as an intermediate CBC-MAC input block $B^{i'}_{j'} \oplus Y^{i'}_{j'-1}$ for $i' < i$ or $i' = i, j' < j$. We say x is of $\mathsf{type3}$ when it has been added as a counter block $\mathsf{ctr}(N^{i'}, \tau_{i'}, j')$ for $i' \le i$.

$\mathsf{Case1}$: $j = 0$. The input B^i_0 to f that may set $\mathsf{coll\text{-}bad}$ has no randomness. We examine collision probabilities with the three types of elements in $\mathrm{Dom}(f)$, determining that the maximal collision probability in this case is $1/2^n$:

$\mathsf{type1}$: As $\mathsf{llcp}_n(i) < j = 0$ implies that N^i has not been used in any previous query, the probability of collision with an x of $\mathsf{type1}$ is 0.

$\mathsf{type2}$: This collision is equivalent to the event $Y^{i'}_{j'-1} = B^i_0 \oplus B^{i'}_{j'}$. Due to the induction assumption, $Y^{i'}_{j'-1}$ is statistically independent of all variables returned to the adversary so far (ciphertext blocks and tags), so this collision happens with probability at most $1/2^n$.

$\mathsf{type3}$: This collision happens with probability zero, as the ranges of the two encoding functions $\mathsf{encB}_0(\cdot,\cdot,\cdot,\cdot)$ and $\mathsf{ctr}(\cdot,\cdot,\cdot)$ have an empty intersection, thanks to the dedicated domain separation bits.

Case2: $0 < j = \mathsf{llcp}_n(i) + 1$. The potentially collision-causing input to f here is $B^i_j \oplus Y^i_{j-1}$, such that the value $Y^i_{j-1} = Y^{\tilde{i}}_{\mathsf{llcp}_n(i)}$ has been sampled in $\tilde{i}^{\text{th}}$ query with $\tilde{i} = \min\{r < i \mid \mathsf{llcp}_n(B^i, B^r) = \mathsf{llcp}_n(i) + 1\}$. Examining the collision probabilities with the three types of elements in $\mathrm{Dom}(f)$ reveals the maximal collision probability in this case to be $1/2^n$:

type1: This is equivalent to $Y^i_{j-1} = B^i_j \oplus B^{i'}_0$. Even though Y^i_{j-1} has been sampled in $\tilde{i}^{\text{th}}$ query, the induction assumption implies that it is statistically independent of all variables seen by the adversary, so happens with probability at most $1/2^n$.

type2: We have two sub-cases here. In the first, special case, the collision event is $Y^i_{j-1} \oplus B^i_j = Y^{i'}_{j-1} \oplus B^{i'}_j$, such that $\mathsf{llcp}_n(B^i, B^{i'}) = \mathsf{llcp}_n(i) + 1$. This implies $Y^i_{j-1} = Y^{i'}_{j-1}$ but since $j > \mathsf{llcp}_n(i)$, we must have $B^i_j \neq B^{i'}_j$ by the definition of the longest common prefix, corresponding to a collision probability zero.

In the second, general case, the collision event $Y^i_{j-1} \oplus B^i_j = Y^{i'}_{j'-1} \oplus B^{i'}_{j'}$ with $j' \neq \mathsf{llcp}_n(i)+1$ happens with probability $1/2^n$, because the variables Y^i_{j-1} and $Y^{i'}_{j'-1}$, even though both sampled in a previous query, are statistically independent, and unknown to $\mathscr{A}$ due to induction hypothesis.

type3: Similarly, as with type1, the collision event $Y^i_{j-1} \oplus B^i_j = \mathsf{ctr}(N^{i'}, \tau_{i'}, j')$ happens with probability at most $1/2^n$.

Case3: $\mathsf{llcp}_n(i)+1 > j \leq \ell_i$. Because of the induction assumption, and because $\mathsf{llcp}_n(i)+1 > j$, the variable Y^i_{j-1} is fresh and has not been used in the game before. The probability of collision with an $x \in \mathrm{Dom}(f)$ of any type is thus $1/2^n$.

As there are at most $2\sigma + 3q$ possible values for (i, j) that could set coll-bad, and because when sampling y^i_j each such value (i, j) we have $|\mathrm{Dom}(f)| \leq 2\sigma + 3q$, we have $\Pr[\mathscr{A}^{\mathbf{G_0}} \text{ sets coll-bad}] \leq (2\sigma + 3q)^2/2^n$.

Bad events. We next define the following fine-grained bad events:

ey-bad(i, j) is defined as the event that the i^{th} adversarial query is an encryption query *and* bad gets set to true when sampling the value y^i_j (i.e., it has remained false in the previous $i-1$ queries and when sampling $y^i_0, \ldots, y^i_{j-1}$) for $1 \leq i \leq q$ and $0 \leq j \leq \ell_i$.

ez-bad(i, j) is defined as the event that the i^{th} adversarial query is an encryption query *and* bad gets set to true when sampling the value z^i_j (i.e., it has remained false in the previous $i-1$ queries, when sampling $y^i_0, \ldots, y^i_{\ell_i}$ and when sampling $z^i_0, \ldots, z^i_{j-1}$) for $1 \leq i \leq q$ and $0 \leq j \leq m_i$.

dz-bad(i, j) is defined as the event that the i^{th} adversarial query is an decryption query *and* bad gets set to true when sampling the value z^i_j (i.e., it has remained false in the previous $i-1$ queries and when sampling $z^i_0, \ldots, z^i_{j-1}$) for $1 \leq i \leq q$ and $0 \leq j \leq m_i$.

dy-bad(i, j) is defined as the event that the i^{th} adversarial query is an decryption query *and* bad gets set to true when sampling the value y^i_j (i.e., it has remained false in the previous $i-1$ queries, when sampling $z^i_0, \ldots, z^i_{m_i}$, and when sampling $y^i_0, \ldots, y^i_{j-1}$) for $1 \leq i \leq q$ and $0 \leq j \leq \ell_i$.

Denoting by E the event "$\mathscr{A}^{\mathbf{G_0}}$ sets coll-bad" further on, we have $\Pr[\mathscr{A}^{\mathbf{G_0}} \text{ sets bad} \mid \neg\mathsf{E}] \leq$

$$\sum_{i=1}^{q} \sum_{j=1}^{\ell_i} \Pr[\text{ey-bad}(i, j) \vee \text{dy-bad}(i.j) \mid \neg\mathsf{E}] + \sum_{j=1}^{m_i} \Pr[\text{ez-bad}(i, j) \vee \text{dz-bad}(i.j) \mid \neg\mathsf{E}] \ . \quad (1)$$

As ey-bad(i,j) and dy-bad$(i.j)$ are mutually exclusive, we have Pr[ey-bad$(i,j) \vee$ dy-bad$(i.j) \mid \neg\mathsf{E}] \leq \max(\Pr[\text{ey-bad}(i,j) \mid \neg\mathsf{E}], \Pr[\text{dy-bad}(i,j) \mid \neg\mathsf{E}])$. Similarly, we have Pr[ez-bad$(i,j) \vee$ dz-bad$(i.j) \mid \neg\mathsf{E}] \leq \max(\Pr[\text{ez-bad}(i,j) \mid \neg\mathsf{E}], \Pr[\text{dz-bad}(i,j) \mid \neg\mathsf{E}])$.

We first turn to the bad events related to the sampling of y_j^i. We have that Pr[ey-bad$(i,j) \mid \neg\mathsf{E}] =$ Pr[dy-bad$(i,j) \mid \neg\mathsf{E}] = 0$. This is because for $1 \leq j \leq \mathsf{llcp}_n(i)$, the input value $Y_{j-1}^i \oplus B_j^i$ has already been used in a previous query i' with $\tau_{i'} = \tau_i$, as the latter is a necessary condition for $\mathsf{llcp}_n(i) > -1$. For $\mathsf{llcp}_n(i) < j \leq \ell_i$, we must have $Y_{j-1}^i \oplus B_j^i \notin \mathrm{Dom}(f)$, so bad cannot be set.

We bound Pr[ez-bad$(i,j) \mid \neg\mathsf{E}]$ which corresponds to $\mathsf{ctr}(N^i, \tau_i, j) \in \mathrm{Dom}(f) \backslash \mathrm{Dom}(f_{\tau_i})$ through a case analysis of the collision probabilities with individual elements of this set, using the three types of domain points defined before.

type1: This event is equivalent to $\mathsf{ctr}(N^i, \tau_i, j) = B_0^{i'}$, such that $\tau_i \neq \tau_{i'}$. Similarly, as with Case1-type3 collision, this event happens with probability 0, thanks to the domain separation of the involved encoding functions.

type2: This is equivalent to $\mathsf{ctr}(N^i, \tau_i, j) = Y_{j'-1}^{i'} \oplus B_{j'}^{i'}$ with $\tau_i \neq \tau_{i'}$. As previously argued, if coll-bad is not set during the experiment, all variables observed by the adversary are statistically independent of $Y_{j'-1}^{i'}$ for $1 \leq i' \leq q$ and $1 \leq j \leq \ell_i$. This event thus happens with probability $1/2^n$.

type3: This event corresponds to $\mathsf{ctr}(N^i, \tau_i j) = \mathsf{ctr}(N^{i'}, \tau_i, j')$ with $\tau_i \neq \tau_{i'}$, which obviously happens with probability zero.

With $|\mathrm{Dom}(f) \backslash \mathrm{Dom}(f_{\tau_i})| \leq 2\sigma + 3q$ at any point in the experiment, we have Pr[ez-bad$(i,j) \mid \neg\mathsf{E}] \leq (2\sigma + 3q)/2^n$. With the decryption event dy-bad$(i,j) \mid \neg\mathsf{E}$ being analyzed analogously, we also have that Pr[ez-bad$(i,j) \vee$ dz-bad$(i.j) \mid \neg\mathsf{E}] \leq (2\sigma + 3q)/2^n$. By back-substituting the upper-bounds into (1), evaluating the sums and maximizing ℓ_i, we get $\Pr[\mathscr{A}^{\boldsymbol{G_0}}$ sets bad $\mid \neg\mathsf{E}] \leq (2\sigma + 3q)^2/2^n$. □

6 Experimental Validation of Energy Efficiency

In many applications relying on battery-operated low-power devices, (such as wireless sensor networks), energy is among the most critical resources. In this section, we experimentally validate that the flexible trade-off between security level and communication overhead enabled by **vCCM**, an nvAE scheme, does indeed translate to measurable energy savings. Our experiment and projections are based on a simple communication scenario where one device always acts as a transmitter and another device as a receiver. We measure and compare the energy consumption of the sending device when using CCM and vCCM, respectively, to encrypt data and transmit it using a sub-GHz transceiver.

Communication scenario. In our scenario, the sender transmits two types of messages, "non-critical" and "critical", periodically. The "non-critical" messages are sent frequently and are assumed to require a lower level of protection. The "critical" messages are sent sporadically and are assumed to require a higher level of security. The sender does not receive any transmissions. This scenario is a simplified model of numerous applications where wireless sensor nodes regularly sense and report on their environment, as for example temperature sensors in smart building systems, parking-lot sensors reporting occupancy, manufacturing-line monitoring sensors in a predictive maintenance system, environmental sensors for prediction of avalanches and so on. Here, the non-critical messages correspond to the sensing data, where the impact of corruption of data due to an occasional forgery is typically low (in the sense of risk management). The critical messages correspond to control traffic (such as reporting a permanent shutdown due to a drained battery), where sporadic

forgeries may have a significant impact. Thus, stretching the non-critical ciphertexts less than the critical ones is appropriate, from the perspective of a cost-security trade-off.

Experimental Setup. Our setup consists of two embedded HW platforms, a *sender* and a *receiver*. The sender is a custom low-power embedded platform designed at CSEM, called Wisenode VXI (WN), with the nRF52840 SoC by Nordic Semiconductors [nrf] as the main micro-controller unit (MCU) and the Sx1261 transceiver by Semtech [sx1] used for wireless communication. The receiver is a Raspberry Pi 3B board [rpi] extended with the LoRa/GPS hat by Dragino [dra]. For the transmission, we send raw packets using the LoRa PHY modulation.

The HW platforms have been selected as follows. The sender platform uses an MCU well-known for its low power consumption, and the LoRa radio is especially well-suited to minimize overhead of transmission in terms of energy consumption. This would be a natural choice for an application that has low-power requirements. The receiver platform is based on widely available components off-the-shelf, relying on open-source libraries, serving as a reference, and indirectly as a validation for the communication stack of the sender.

To determine the energy consumption of the sender, we powered the sender with a lab power supply at 3V, and measured the immediate current from the power supply with a Keysight CX3324A Device Current Waveform Analyzer using a passive 0Ω probe at a sampling frequency of 100MHz.

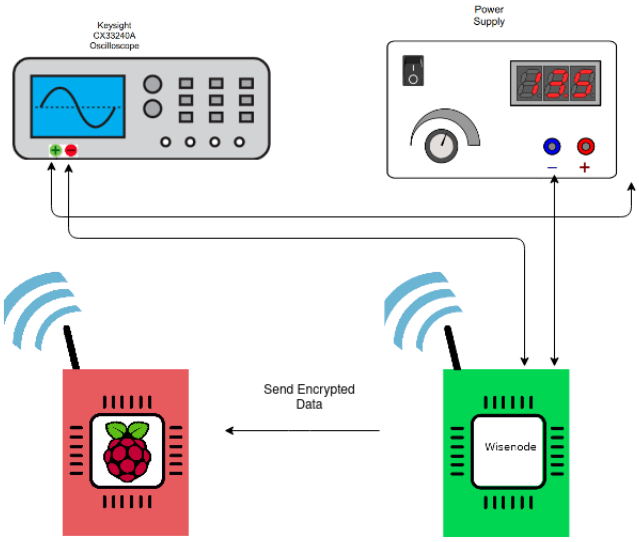
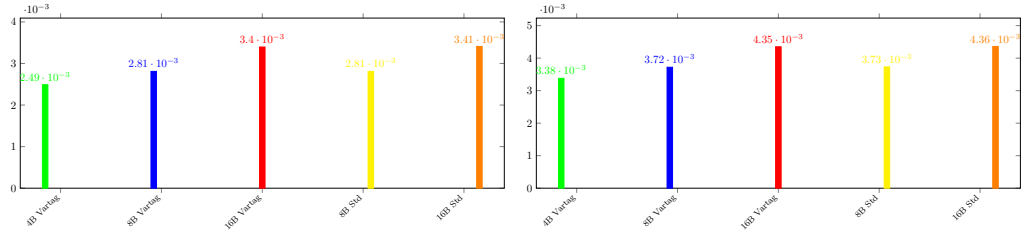


Figure 10: Power consumption measurement setup.

Implementation. The sender is running CSEM's custom embedded real-time, multi-process operating system μ111 [MFI94, MFG99], which implements most of the embedded and process synchronization primitives, such as semaphores and precise timers, and is designed specifically for low-power applications. Thanks to these features, μ111 lends itself well for the task at hand. For the experiment, a driver for the Sx1261 radio has been integrated in the OS. The application code uses a hardware timer run in one process to precisely schedule data encryption and transmissions, in a second, synchronized process. All unneeded peripherals are disabled, and the MCU and the radio are in sleep mode between transmissions. The receiver is running Raspbian, the native, Linux-based OS for Rasperry Pi, with the default driver for the Dragino LoRa hat. Both the sender and

receiver used the same implementation of **vCCM**, based on the CCM implementation of mbedTLS [mbe].

Measurements. We carry out an experiment consisting of 10 measurements. In each measurement, we capture the immediate current drawn by the sender, while repeatedly encrypting and sending a payload every 10 seconds, acquiring data from 70 transmissions. Such a measurement is done for each combination of a plaintext length (4 or 16 bytes), and AE-scheme tag-length pair (CCM and 8 byte tags, CCM and 16-byte tags, **vCCM** and 4-byte tags, **vCCM** and 8-byte tags, and **vCCM** and 16-byte tags). In a post-processing scripts, we isolate the consumption "peak" corresponding to the entire wake-up time of the sender during each of the 70 transmissions. This includes the consumption due to encryption, transmission and the OS-incurred processing overhead. In each measurement, we compute the average duration and average energy consumed[3] per single transmission "peak". These average values are displayed in Figure 12 and visualized in Figure 11.



(a) Energy Consumption(4B Messages) **(b)** Energy Consumption(16B Messages)

Figure 11: Experiments Results

Tag length, AE scheme	Avg Duration per Transmission(s)/4B	Avg Energy Consumption per Transmission(J)/4B	Avg Duration per Transmission(s)/16B	Avg Current Energy per Transmission(J)/16B
4B **vCCM**	0.04106	0.002489	0.05608	0.00338
8B **vCCM**	0.04639	0.00281	0.06164	0.00372
16B **vCCM**	0.05608	0.003396	0.07196	0.004349
8B CCM	0.04639	0.00281	0.06173	0.00373
16B CCM	0.05627	0.00341	0.07196	0.00436

Figure 12: Experimental result for 4-bytes and 16-bytes messages respectively

Energy Savings Projections. The energy consumption for transmissions with different values of stretch already shows that varying the stretch does result in a tangible energy economy. To better approximate the savings as perceived by the user of AE, we further investigate the *overall* energy economy (i.e., also taking into account the sleep current). We estimate it by making a projection of energy consumption of the sender for several instances of our communication scenario.[4]

To compute a projected energy consumption, we assume the sender transmits a non-critical message every 10 seconds, and a critical message once every minute. When using **vCCM**, the non-critical ciphertexts are stretched less than the critical ciphertexts, such that we consider

[3] An approximation computed as the sum of current samples times the voltage times the inverse of sampling frequency.

[4] The reason for doing a projection rather than a direct measurement is that the sender platform was not yet fully optimized at the time of writing of this paper, and its sleep current was an order of magnitude higher than what it is expected to achieve.

several combinations of plaintext length and two tag lengths. When using CCM, we treat all plaintexts with the longer of the two tag lengths.

The approximate energy consumed by a T-second long operation of the sender using the standard CCM $E_{\mathrm{CCM}}(T)$ is computed as shown in (2), where $\bar{I}$ denotes the average current drawn during a transmission (can be used and computed from Figure 12 w.l.o.g.) and $\bar{t}$ denotes the average duration of the wake-up state per transmission. The formula assumes an optimal idle state current consumption of 18 μA and that the voltage is 3V.

$$3 \cdot \left((T - \lfloor T/10 \rfloor \cdot \bar{t}) \cdot 18 \times 10^{-6} + \lfloor T/10 \rfloor \cdot \bar{t} \cdot \bar{I}\right) \text{ J} . \tag{2}$$

The approximate energy consumed by a T-second long operation of the sender using **vCCM** $E_{\mathbf{vCCM}}(T)$ is computed as shown in (3), where $\bar{I_1}$ and $\bar{I_2}$ denote the average current drawn during transmission of the non-essential and essential messages respectively, and $\bar{t_1}$ and $\bar{t_2}$ denote the average duration of wake-up state per transmission for non-essential and essential messages respectively. The formula again assumes an optimal idle state current consumption of 18 μA and that the voltage is 3V.

$$3 \cdot \left(T - \lfloor T/10 \rfloor \cdot \left(\frac{5}{6} \cdot \bar{t_1} + \frac{1}{6} \cdot \bar{t_2}\right)\right) \cdot 18 \times 10^{-6} + \lfloor T/10 \rfloor \cdot \left(\frac{5}{6} \cdot \bar{t_1} \cdot \bar{I_1} + \frac{1}{6} \cdot \bar{t_2} \cdot \bar{I_2}\right) \text{ J} . \tag{3}$$

By evaluating the formulas (2) and (3) using the experimental results from Table 12 and by setting the time $T = 3600$s (1 hour), we obtain the results displayed in Table 13, and visualized in Figure 14 for $T \leq 3600$s.

From Table 13, we see that we obtain 8% of overall energy saving when using 4B/8B tags with **vCCM** instead of the 8B tags with CCM, 21% energy savings when using the 4B/16B tags with **vCCM** instead of the 16B tags with CCM, and 14% energy savings when using the 8B/16B tags with **vCCM** instead of 16B tags with CCM. This shows that an optimization as simple as adjusting the tag length in communication scenarios with modest frame sizes leads to a noticeable decrease of energy consumption. The relevance for practical applications is even higher due to the fact that these savings are almost for free, owing to the simplicity of the black-box transform underlying **vCCM**.

	Energy Savings	Tag Length (Non-Critical/Critical)	Projected Energy Consumption (J)	
			4B messages	16B Messages
(1)	8% less than (4)	4B/8B Vartag CCM	0.30902	0.40516
(2)	21% less than (5)	4B/16B Vartag CCM	0.31981	0.42018
(3)	15% less than (5)	8B/16B vartag CCM	0.34532	0.45601
(4)	-	8B standard CCM	0.36434	0.44187
(5)	-	16B standard CCM	0.40053	0.53184

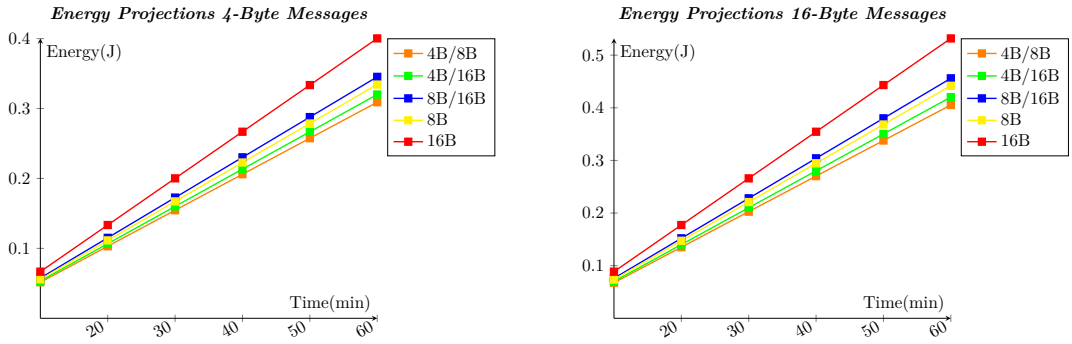
Figure 13: Projections Energy consumption projections for 1-hour operation

7 Yoga Is Not For Everyone

In Sections 4 and 5 we show that it is possible for an nAE secure scheme to be black-box transformed into an nvAE secure scheme. This, together with the potential for substantial energy savings achievable with help of nvAE schemes (showed in Section 6), raises the following questions.

Does this transformation work for all nAE schemes? This question has previously been answered negatively by Reyhanitabar, Vaudenay and Vizár, who showed that an entire class of nAE secure schemes[5] succumbs to a non-trivial nvAE forgery attack when they are transformed into nvAE schemes by encoding the tag length into their nonce, or into their AD, or even into both their nonce and their AD simultaneously [RVV16]. Moreover, this class of AE schemes includes OCB [RBBK01, Rog04, KR11] and GCM [MV04], two very well known and widely used

[5] More precisely this weakness applies to all schemes that internally use ciphertext-translation method to authenticate AD [Rog02].



(a) 4-byte messages. **(b)** 16-byte messages.

Figure 14: Projections of energy consumption.

constructions, which further highlights CCM as the target whose transformation has the best potential for *immediate* real-world impact.

Does this transformation work for any other nAE schemes? We confidently conjecture an affirmative response to this question. The insight learned from the forgery attack presented by Reyhanitabar et al. (and in a less general version by the Ascon team in the CAESAR competition before that [Eic]) is, informally speaking, that for an nvAE construction to be secure, each call to the underlying cryptographic primitive must have a stretch-dependent input. The blakcbox-transformed OCB and GCM do not have this property because the processing of their AD blocks is parallel and independent of the nonce. Interestingly enough, the sequential nature of CCM's authentication tag computation, sometimes criticized for its inefficiency [RW03], provides nvAE security with the nonce-based transform: the nonce is part of the input to the first blockcipher call in the plain CBC MAC, which propagates the "influence" of stretch throughout the computation of the authentication tag.

This observation immediately suggests sequential modes as good candidates for the nAE-to-nvAE transform from Section 4. In particular, sponge-based AE modes can be safely conjectured to enjoy nvAE security with tag length encoded in the nonce. The informal proof sketch is that, thanks to the reduction to a duplex object [BDPA11, MRV15] (essentially a blockwise, stateful PRF), queries with different tag lengths will have all internal sponge states sampled "independently", which yields the **kess** property. A formal proof is required to confirm this intuition, however. Moreover, sponge variants that cannot be reduced to keyed duplex, such as Ascon [DEMS], will require dedicated proofs.

Reyhanitabar et al. [RVV16] also presented vΘCB, an nvAE secure variant of OCB, where the tag length has been included as a tweak component for the underlying tweakable blockcipher, in order to obtain the **kess** property. This suggests tweakable blockcipher-based nAE modes as candidates for the nAE-to-nvAE transform from Section 4, though this class of AE schemes seems to be without a significant representative construction.

8 Discussion

We have presented the first nvAE secure scheme that is obtained as a truly black-box transform of a previous AEAD construction, such that the latter is perhaps the most widely supported AEAD standard in embedded computational platforms. We have then experimentally confirmed that the use of such a scheme is of practical interest, and brings measurable improvements of efficiency.

One important question, which has been addressed only partially in the existing literature, is the resistance of CCM to multiple forgeries; i.e., how difficult is it to mount a forgery with a τ-bit tag given that the adversary has already succeeded in making one or more forgeries? This is especially relevant when considering extreme tag lengths (below 32 bits used e.g., in Bluetooth standard), which can be meaningful for certain applications. Forler et al. indicate that CCM does resist to such attacks in nonce-respecting setting [FLLW17], suggesting vCCM may be used with

extremely short tags. However, it is necessary to integrate the reforgeability and the **nvae** security notions and investigate the corresponding relations among notions to fully analyze the impact of simultaneous tag variations and reforgeries on the security of vCCM and other nvAE schemes.

Another interesting open question is to identify which NIST LWC candidates are eligible for an nvAE black-box transform, defining and analyzing these transforms, in order to compensate for the lack of consideration for variable stretch in the standardization project.

Acknowledgments

We would like to thank Jean-Marc Koller, Robin Berguerand and Lorenzo Bergamini for their help with the experimental setup.

References

[ADP+20] Elena Andreeva, Arne Deprez, Jowan Pittevils, Arnab Roy, Amit Singh Bhati, and Damian Vizár. New Results and Insighs on ForkAE. *NIST LWC workshop*, 2020.

[ALP+19] Elena Andreeva, Virginie Lallemand, Antoon Purnal, Reza Reyhanitabar, Arnab Roy, and Damian Vizár. Forkcipher: A New Primitive for Authenticated Encryption of Very Short Messages. In Steven D. Galbraith and Shiho Moriai, editors, *ASIACRYPT 2019, Proceedings, Part II*, volume 11922 of *LNCS*, pages 153–182. Springer, 2019.

[BDPA11] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. Duplexing the Sponge: Single-Pass Authenticated Encryption and Other Applications. In Ali Miri and Serge Vaudenay, editors, *SAC 2011, Revised Selected Papers*, volume 7118 of *LNCS*, pages 320–337. Springer, 2011.

[BP17] Alex Biryukov and Leo Perrin. State of the Art in Lightweight Symmetric Cryptography. Cryptology ePrint Archive, Report 2017/511, 2017. https://eprint.iacr.org/2017/511.

[BR06] Mihir Bellare and Phillip Rogaway. The Security of Triple Encryption and a Framework for Code-Based Game-Playing Proofs. In Serge Vaudenay, editor, *EUROCRYPT 2006, Proceedings*, volume 4004 of *LNCS*, pages 409–426. Springer, 2006.

[DEMS] Christoph Dobraunig, Maria Eichlseder, Florian Mendel, and Martin Schl affer. Ascon v1.1 Submission to the CAESAR Competition. https://competitions.cr.yp.to/round2/asconv11.pdf.

[dra] Lora/GPS HAT. http://wiki.dragino.com/index.php?title=Lora/GPS_HAT. Accessed: 2021-02-23.

[DWF03] Russ Housley Doug Whiting and Niels Ferguson. Counter with CBC-MAC. LNCS, 2003.

[Eic] Maria Eichlseder. Remark on variable tag lengths and OMD. `crypto-competitions` mailing list. April 25, 2014.

[FLLW17] Christian Forler, Eik List, Stefan Lucks, and Jakob Wenzel. Reforgeability of Authenticated Encryption Schemes. In Josef Pieprzyk and Suriadi Suriadi, editors, *ACISP 2017, Proceedings, Part II*, volume 10343 of *LNCS*, pages 19–37. Springer, 2017.

[GS19] Sebati Ghosh and Palash Sarkar. Variable Tag Length Message Authentication Code Schemes. *IACR Cryptology ePrint Archive*, 2019:1347, 2019.

[Jon02] Jakob Jonsson. On the Security of CTR + CBC-MAC. In Kaisa Nyberg and Howard M. Heys, editors, *SAC 2002*, volume 2595 of *LNCS*, pages 76–93. Springer, 2002.

[KM] Masanobu Katagi and Shiho Moriai. Lightweight Cryptography for the Internet of Things. https://iab.org/wp-content/IAB-uploads/2011/03/Kaftan.pdf.

[KR11] Ted Krovetz and Phillip Rogaway. The Software Performance of Authenticated-Encryption Modes. In Antoine Joux, editor, *FSE 2011*, volume 6733 of *LNCS*, pages 306–327. Springer, 2011.

[mbe] arm MBED. https://tls.mbed.org. Accessed: 2021-02-23.

[MFG99] Francesco Mondada, Edoardo Franzi, and André Guignard. The Development of Khepera. In *Proceedings of the First International Khepera Workshop*, HNI-Verlagsschriftenreihe, Heinz Nixdorf Institut, 64, pages 7–14, 1999.

[MFI94] Francesco Mondada, Edoardo Franzi, and Paolo Ienne. Mobile robot miniaturisation: A tool for investigation in control algorithms. In *Experimental Robotics III*, pages 501–513. Springer, 1994.

[MRV15] Bart Mennink, Reza Reyhanitabar, and Damian Vizár. Security of Full-State Keyed Sponge and Duplex: Applications to Authenticated Encryption. In Tetsu Iwata and Jung Hee Cheon, editors, *ASIACRYPT 2015, Proceedings, Part II*, volume 9453 of *LNCS*, pages 465–489. Springer, 2015.

[MV04] David A. McGrew and John Viega. The Security and Performance of the Galois/Counter Mode (GCM) of Operation. In Anne Canteaut and Kapalee Viswanathan, editors, *INDOCRYPT 2004, Proceedings*, volume 3348 of *LNCS*, pages 343–355. Springer, 2004.

[nisa] Lightweight Cryptography. National Institute of Standards and Technology. https://csrc.nist.gov/Projects/lightweight-cryptography.

[nisb] Submission Requirements and Evaluation Criteria for the Lightweight Cryptography Standardization Process. National Institute of Standards and Technology. https://csrc.nist.gov/CSRC/media/Projects/Lightweight-Cryptography/documents/final-lwc-submission-requirements-august2018.pdf.

[nrf] nRF52840 Product Specification v1.2. Nordic Semiconductor. https://infocenter.nordicsemi.com/index.jsp?topic=%2Fps_nrf52840%2Fkeyfeatures_html5.html.

[NRS14] Chanathip Namprempre, Phillip Rogaway, and Thomas Shrimpton. Reconsidering Generic Composition. In Phong Q. Nguyen and Elisabeth Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 257–274. Springer, 2014.

[RBBK01] Phillip Rogaway, Mihir Bellare, John Black, and Ted Krovetz. OCB: A Block-Cipher Mode of Operation for Efficient Authenticated Encryption. In *ACM CCS 2001*, pages 196–205, 2001.

[Rog02] Phillip Rogaway. Authenticated-Encryption with Associated-Data. In *ACM CCS 2002*, pages 98–107, 2002.

[Rog04] Phillip Rogaway. Efficient Instantiations of Tweakable Blockciphers and Refinements to Modes OCB and PMAC. In Pil Joong Lee, editor, *ASIACRYPT 2004*, volume 3329 of *LNCS*, pages 16–31. Springer, 2004.

[rpi] Raspberry Pi 3 Model B. Raspberry Pi Foundation. https://www.raspberrypi.org/products/raspberry-pi-3-model-b/.

[RS06] Phillip Rogaway and Thomas Shrimpton. A Provable-Security Treatment of the Key-Wrap Problem. In Serge Vaudenay, editor, *EUROCRYPT 2006, Proceedings*, volume 4004 of *LNCS*, pages 373–390. Springer, 2006.

[RVV16] Reza Reyhanitabar, Serge Vaudenay, and Damian Vizár. Authenticated Encryption with Variable Stretch. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *ASIACRYPT 2016, Proceedings, Part I*, volume 10031 of *LNCS*, pages 396–425, 2016.

[RW03] Phillip Rogaway and David Wagner. A Critique of CCM. *IACR Cryptology ePrint Archive*, 2003:70, 2003.

[SLD17] Reihaneh Safavi-Naini, Viliam Lisý, and Yvo Desmedt. Economically optimal variable tag length message authentication. In Aggelos Kiayias, editor, *Financial Cryptography and Data Security - FC 2017*, volume 10322 of *LNCS*, pages 204–223. Springer, 2017.

[SWE02] Sanjay E. Sarma, Stephen A. Weis, and Daniel W. Engels. RFID Systems and Security and Privacy Implications. In Burton S. Kaliski Jr., Çetin Kaya Koç, and Christof Paar, editors, *CHES 2002*, volume 2523 of *LNCS*, pages 454–469. Springer, 2002.

[sx1] Semtech SX1261: LoRa Core™ Long Range Low Power LoRa® RF Transceiver. Semtec. https://www.semtech.com/products/wireless-rf/lora-core/sx1261.

[WHF02] D. Whiting, R. Housley, and N. Ferguson. AES Encryption & Authentication Using CTR Mode & CBC-MAC. IEEE P802.11 doc 02/001r2, May 2002.

[WHF03] D. Whiting, R. Housley, and N. Ferguson. Counter with CBC-MAC (CCM). IETF RFC 3610 (Informational), September 2003.

[WSRE03] Stephen A. Weis, Sanjay E. Sarma, Ronald L. Rivest, and Daniel W. Engels. Security and Privacy Aspects of Low-Cost Radio Frequency Identification Systems. In Dieter Hutter, Günter Müller, Werner Stephan, and Markus Ullmann, editors, *Security in Pervasive Computing, First International Conference*, volume 2802 of *LNCS*, pages 201–212. Springer, 2003.