Authenticated Encryption with Variable Stretch

Reza Reyhanitabar[1], Serge Vaudenay[2], and Damian Vizár[2]

[1] NEC Laboratories Europe, Germany
[2] EPFL, Switzerland

May 19, 2016

Abstract. In conventional authenticated-encryption (AE) schemes, the *ciphertext expansion*, *a.k.a. stretch* or *tag length*, is a constant or a parameter of the scheme that must be *fixed* per key. However, using variable-length tags per key can be desirable in practice or may occur as a result of a misuse. The RAE definition by Hoang, Krovetz, and Rogaway (Eurocrypt 2015), aiming at the *best-possible* AE security, supports variable stretch among other strong features, but achieving the RAE goal incurs a particular inefficiency: *neither encryption nor decryption can be online*. The problem of enhancing the well-established nonce-based AE (nAE) model and the standard schemes thereof to support variable tag lengths per key, without sacrificing any desirable functional and efficiency properties such as *online* encryption, has recently regained interest as evidenced by extensive discussion threads on the CFRG forum and the CAESAR competition. Yet there is a lack of formal definition for this goal. First, we show that several recently proposed heuristic measures trying to augment the known schemes by inserting the tag length into the nonce and/or associated data *fail* to deliver any meaningful security in this setting. Second, we provide a formal definition for the notion of nonce-based variable-stretch AE (nvAE) as a natural extension to the traditional nAE model. Then, we proceed by showing a second modular approach to formalizing the goal by combining the nAE notion and a new property we call *key-equivalent separation by stretch* (*kess*). It is proved that (after a mild adjustment to the syntax) any nAE scheme which additionally fulfills the *kess* property will achieve the nvAE goal. Finally, we show that the nvAE goal is efficiently and provably achievable; for instance, by simple tweaks to off-the-shelf schemes such as OCB.

Keywords: Authenticated encryption, variable-length tags, robustness, security definitions, CAESAR competition.

1 Introduction

Authenticated encryption (AE) algorithms have recently faced an immense increase in popularity as appropriate cryptographic tools for providing *data* confidentiality (privacy) and integrity (together with authenticity) services simultaneously. The notion of AE, as a cryptographic scheme in its own right, was originally put forward in [2,3,12] and further evolved to notions of nonce-based AE (nAE) [25], nonce-based AE with associated data (AEAD) [22,24], deterministic AE (DAE) and misuse-resistant AE (MRAE) [26], online nonce-misuse resistant AE [7], AE under the release of unverified plaintext (AE-RUP) [1], robust AE (RAE) [9], and online AE (OAE2) [10].

Providing *authenticity* requires any AE scheme to incur a non-zero ciphertext expansion or stretch, $\tau = |C| - |M|$, where $|M|$ and $|C|$ are the lengths of the plaintext and ciphertext in bits, respectively. Most standard AE schemes adopt a syntax in which the ciphertex is explicitly partitioned as $C = C_{\texttt{core}}||\texttt{Tag}$ with $C_{\texttt{core}}$ as the ciphertext core (decryptable to a putative plaintext) and $\texttt{Tag}$ as the authentication tag (used for verifying the decrypted message). In this paper, we will use the terms *ciphertext expansion*, *stretch* and *tag length* interchangeably unless the syntax of an AE scheme (e.g. an RAE scheme) does not allow partitioning of the ciphertext to a core and a tag part, in which case we use the general term *stretch*.

The problem. This paper investigates the problem of using an AE scheme with variable-length tags (variable stretch) under the same key. All the known security notions for AE schemes [1,7,10,22,24,26] and constructions thereof, with the exception of RAE [9], assume that the stretch τ is a constant or a scheme parameter which must be *fixed* per key, and security is proved under this assumption. A correct usage of such a scheme shall ensure that two instances of the same scheme with different stretches τ_1 and τ_2 always use two independently chosen keys K_1 and K_2. However, this rigid correct-use mandate may be violated in practice for different reasons.

First, AE schemes may be used with variable-length tags per key due to misuse and poorly engineered security systems. Prior "Disasters" [5] have shown that it's a question of when, not if, a misuse will eventually happen in applications of (symmetric-key) cryptographic schemes in practice.

The ongoing CAESAR competition [4] has explicitly listed a set of conventional confidentiality and integrity goals for AE, but has left "any additional security goals and robustness goals that the submitters wish to point out" as an option. Among the potential additional goals, *robustness* features, in particular, different flavors of misuse-resistance to nonce reuse [7,26] have attracted a lot of attention. While the recent focus has been mainly on nonce misuse, proper characterization and formalization of other potential misuse dimensions seems yet a challenge to be further investigated. The current literature lacks a systematic approach to formalizing an appropriate notion of AE with misuse-resistance to tag-length variation under the same key, *without sacrificing* interesting functional and efficiency features such as online encryption.

Second, there are use cases such as resource-constrained communication devices, where the support for variable-length tags is desired, but changing the key per tag length and renegotiating the system parameters is a costly process due to bandwidth and energy constraints. In those cases, supporting variable stretch per key while still being able to provide a "sliding scale" authenticity is deemed to be a useful functional and efficiency feature [29].

The problem has appeared to be highly interesting from both theoretical and practical perspectives as evidenced by the relatively long CFRG forum thread on issues arising from variable-length tags in OCB [14], followed by ongoing discussions in the CAESAR competition mailing list [11], which in turn has motivated several second-round CAESAR candidates to be tweaked [11, 15, 18] with the aim of providing some *heuristic* measures for addressing the problem.

Issues arising from variable stretch per key. Lack of support for variable-length tags per key in conventional AE models, in particular in the widely-used nAE security model, is not just a theoretical and definitional complaint, rather all known standard AE schemes such as the widely-deployed CCM, GCM, and OCB schemes do *misbehave* in one way or another if misused in this way [14,21,28]. Depending on the application scenario, the consequences of such a misbehavior may range from a degraded security level to a complete loss of security.

A CFRG forum discussion thread initiated by Manger [14], has raised the following concerns with an "Attacker changing tag length in OCB":

- OCB with different tag lengths are defined. Under the same key, shorter tags are simply truncation of longer tags. The tag length is not mixed into the ciphertext as it never affects any input to the underlying blockcipher. Consequently, given a valid output from e.g. the OCB algorithm with 128-bit tag it is trivial to produce a valid output for the OCB algorithm with 64-bit tag under the same key, by just dropping the last 8 bytes.
- An attacker wanting to change the associated data while keeping the same plaintext and the same tag length as applied by the originator (e.g. 128 bits) only has to defeat the shortest accepted tag length (e.g. 64 bits) and the differences between accepted tag lengths up to the targeted stretch. This is not fulfilled by OCB.
- Would OCB be better if the algorithms with different tag lengths could not affect each other? Perhaps restricting the nonce to <126 bits (instead of <128 bits) and encoding the tag length in 2 bits.

The CFRG discussions concluded by adopting Manger's suggested heuristic measure by designers of OCB: "just drop the tag length into the nonce" [21]. One may call this method *nonce stealing* for tag length akin to "nonce stealing" for associated data (AD), proposed by Rogaway [22] to convert an AE scheme to an AEAD scheme. The problem of variable-length tags per key has regained interest in recent CAESAR competition discussions. Nandi [17] has raised the question whether including the tag length in the associated data can resolve the problem. A natural extension would be combining both measures, i.e., including the tag length as part of both the nonce and the associated data.

But in the absence of a definitional and provable-security treatment of the problem of robustness to tag-length variation per key, the proposed heuristic measures and claims for added security in the tweaked schemes are informal, and only limited to showing lack of some specific type of misbehavior by the schemes.

RAE solves the problem, do we need another definition? RAE aims to capture the "*best-possible*" AE security [9]. Similar to the MRAE and Pseudorandom Injection (PRI) notions [26] it targets robustness to nonce-misuse, but it also improves upon the prior notions by supporting variable stretch and hence sliding scale authenticity for any arbitrary stretch. However, the cost to pay for achieving such a strong goal is that any RAE scheme incurs a particular inefficiency: *neither encryption nor decryption can be online*. We also note that designing an *efficient* RAE scheme, e.g. AEZ [9], essentially entails designing an *efficient* tweakable block cipher with variable-length messages and tweaks at the first place followed by employing it in the encode-then-encipher paradigm, a task that has turned out to be non-trivial as evidenced by several non-ideal properties determined by recent attacks against the core cipher of prior AEZ versions [8].

While RAE aims to facilitate the use of any stretch, even a small one, and promises to provide the best-possible security for any stretch even under nonce-reuse, our main aim in this paper is to provide an enhancement to the

conventional AE models, in particular the popular nAE model, that just adds robustness to tag-length variation under the same key without sacrificing the highly desired *online-ness* feature. Unlike the RAE notion our aim is neither to facilitate/encourage using arbitrarily short tags nor to add nonce-misuse resistance to a scheme which does not already possess such a property. The core goal is to minimize/cut the interferences between instances of an AE scheme (e.g. OCB) using different tag lengths under the same key and to meaningfully achieve the best-possible authenticity in this setting without affecting/damaging the privacy property.

Intuitively, one aims to have an AE scheme that can guarantee τ_c-bit authenticity to the recipient whenever a received ciphertext has a τ_c-bit tag (τ_c-bit stretch) irrespective of adversarial access to other instances of the same algorithm under the same key but different (shorter or longer) τ-bit tags.

Heuristic Measures Fail. We show in Section 3 that *in general*, several recently proposed heuristic measures, such as inserting the tag length into the nonce [21], into the associated data [17] or both methods combined, fail to capture the aforementioned intuition of a meaningful security in the variable-length tag setting. This is done by showing generic forgery attacks against these measures in a large class of nAE schemes (including e.g. GCM and OCB) that follow the "ciphertext translation" design paradigm of Rogaway [22]. The attacks have a much lower verification query complexity for τ bits of stretch than 2^τ. For example, an adversary having access to the instances of the same algorithm with 32-bit, 64-bit, 96-bit and 128-bit tags under the same key will only need a query complexity $O(2^{32})$ to forge a message with a 128-bit tag. The attacks are rather straightforward generalization of the tag-length misusing attack presented by the Ascon team on OMD version 1 [6].

Our Results. We formalize a security notion for nonce-based variable-stretch AE (nvAE). First we provide an all-in-one security definition to formulate the notion. Then we take an alternative modular approach for defining the notion by introducing a property, named *key-equivalent separation by stretch* (**kess**), that together with the conventional nAE security implies the nvAE security notion. While the former approach provides an easy-to-understand, stand-alone definition by directly capturing the whole aim of nvAE, the latter modular approach is easier to work with, at least for proving schemes nvAE-secure, in particular, when one tweaks an existing nAE-secure scheme and wants to establish the nvAE-security of the modified scheme by just proving its **kess** property rather than having to prove everything from scratch. We show that the nvAE goal is efficiently and provably achievable by application of simple tweaks to off-the-shelf popular schemes such as OBC without sacrificing their desirable functional and efficiency features such as online encryption. Furthermore, we establish the relations (implications and separations) between different security notions in the conventional fixed-stretch AE setting and variable-stretch AE setting. A summary of the relations is depicted in Fig. 1.

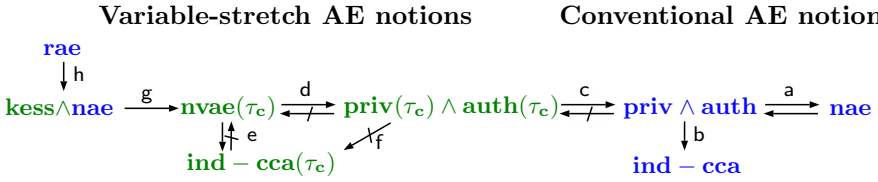

Fig. 1: Relations among notions for nonce-based AE with and without variable stretch. Previous works: a [26], b [2]. This paper: c (Remark 3, attacks in Section 3), d (Remark 3, Corollary 1), e (Theorem 1, Remark 2), f (Proposition 1), g (Theorem 2), h (Remark 4 together with [9]).

Organization of the paper. In Section 2 we overview some of the prior AE definitions. Section 3 describes generic forgery attacks showing ineffectiveness of the heuristic measures of including the tag length in the nonce and/or associated data of a given nAE scheme to support variable-length tags per key. In Section 4 we provide formal definitions for the goal of AE with variable stretch per key and in Section 5 we show how to efficiently achieve the goal. Section A provides some discussions and remarks on the interpretation of the results of this work.

2 Preliminaries and Prior AE Definitions

Resource-parameterized adversarial advantage. The insecurity of a scheme Π in regard to a security property **xxx** is measured using the resource parameterized function $\mathbf{Adv}^{\mathbf{xxx}}_{\Pi}(\mathbf{r}) = \max_{\mathscr{A}}\{\mathbf{Adv}^{\mathbf{xxx}}_{\Pi}(\mathscr{A})\}$, where the maximum is taken over all adversaries $\mathscr{A}$ which use resources bounded by $\mathbf{r}$.

Blockciphers and Tweakable Blockciphers. Let $\mathrm{Perm}(n)$ be the set of all permutations over n-bit strings. Let $\mathrm{Perm}^{\mathcal{T}}(n) \subseteq \{\widetilde{\pi} : \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n\}$ be the set of all functions, s.t. for every $\widetilde{\pi} \in \mathrm{Perm}^{\mathcal{T}}(n)$, $\widetilde{\pi}(t,\cdot)$ is a permutation for every $t \in \mathcal{T}$ where $\mathcal{T}$ is a set of tweaks. We use $\widetilde{\pi}^t(\cdot)$ and $\widetilde{\pi}(t,\cdot)$ interchangeably. Let $E : \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ be a blockcipher and let $\widetilde{E} : \mathcal{K} \times \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$ be a tweakable blockcipher with a non-empty, finite $\mathcal{K} \subseteq \{0,1\}^*$. Let D and $\widetilde{D}$ denote the inverses of E and $\widetilde{E}$ respectively. Let $E_K(\cdot) = E(K,\cdot)$ and $\widetilde{E}^t_K(\cdot) = \widetilde{E}(K,t,\cdot)$. Let $\mathscr{A}$ be an adversary. Then:

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

$$\mathbf{Adv}^{\pm\widetilde{\mathrm{prp}}}_{\widetilde{E}}(\mathscr{A}) = \Pr\left[K \leftarrow\!\$\ \mathcal{K} : \mathscr{A}^{\widetilde{E}_K, \widetilde{D}_K} \Rightarrow 1\right] - \Pr\left[\widetilde{\pi} \leftarrow\!\$\ \mathrm{Perm}^{\mathcal{T}}(n) : \mathscr{A}^{\widetilde{\pi},\widetilde{\pi}^{-1}} \Rightarrow 1\right]$$

The resource parameterized advantage functions are defined accordingly, considering that the adversarial resources of interest here are the time complexity (t) of the adversary and the total number of queries (q) asked by the adversary.

In the following we recall the security notions for nonce-based AE (nAE) schemes with associated data (a.k.a. "AEAD" schemes) [22] and RAE schemes. We will simply use nAE to refer to any (nonce-based) AEAD scheme as all nAE schemes must now support associated data processing.

Syntax. We augment the syntax of nAE schemes from [22] to include 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 $\mathcal{D} : \mathcal{K} \times \mathcal{N} \times \mathcal{A} \times \mathbb{N} \times \mathcal{C} \to \mathcal{M} \cup \{\perp\}$ are the encryption and decryption algorithm respectively, both deterministic and stateless. We have 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 insist that if $M \in \mathcal{M}$ then $\{0,1\}^{|M|} \subseteq \mathcal{M}$ (any reasonable AE scheme would certainly have this property). We additionally limit ourselves to *correct* and *tidy* schemes [16] with *variable stretch*. Namely, the correctness means that 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 $\mathcal{D}(K, N, A, \tau, C) = M$, and 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*.

Variations in Syntax. In the case of conventional nAE schemes, the expansion of ciphertexts is fixed to some constant value τ; this is equivalent to setting $\mathcal{I}_T = \{\tau\}$. For such schemes, we omit stretch from the list of input arguments of both the encryption and the decryption algorithm. We sometimes create an ordinary nonce-based AE scheme Π' from a nonce-based AE scheme with variable stretch Π by fixing the expansion value for all queries to some value $\tau \in \mathcal{I}_T$. We will denote this as $\Pi' = \Pi[\tau]$.

Two-requirement security definition. The nAE notion was originally formalized by a two-requirement (privacy and authenticity) definition [3, 22]. The privacy of a scheme Π is captured by its indistinguishability from a random strings-oracle in a chosen plaintext attack with non-repeating nonces, while its authenticity is defined as adversary's inability to *forge* a new ciphertext, i.e. issue a decryption query returning $M \neq \perp$. The **priv** advantage of an adversary $\mathscr{A}$ against Π is defined as $\mathbf{Adv}^{\mathbf{priv}}_{\Pi}(\mathscr{A}) = \Pr[\mathscr{A}^{\mathbf{priv\text{-}R}_{\Pi}} \Rightarrow 1] - \Pr[\mathscr{A}^{\mathbf{priv\text{-}I}_{\Pi}} \Rightarrow 1]$ and the **auth** advantage of $\mathscr{A}$ as $\mathbf{Adv}^{\mathbf{auth}}_{\Pi}(\mathscr{A}) = \Pr[\mathscr{A}^{\mathbf{auth}_{\Pi}} \text{ forges}]$ where the corresponding security games are defined in Figure 2. In the following $x \leftarrow\!\$\ \mathcal{S}$ will denote sampling an element x from a set $\mathcal{S}$ with uniform distribution.

All-in-one security definition. Rogaway and Shrimpton introduced an alternative, all-in-one approach for defining the nAE security, and proved it to be equivalent to the two-requirement definition [26]. The all-in-one **nae** notion 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 $\mathbf{Adv}^{\mathbf{nae}}_{\Pi}(\mathscr{A}) = \Pr[\mathscr{A}^{\mathbf{nae\text{-}R}_{\Pi}} \Rightarrow 1] - \Pr[\mathscr{A}^{\mathbf{nae\text{-}I}_{\Pi}} \Rightarrow 1]$ where the corresponding security games are defined in Figure 3.

Robust AE. As mentioned in Section 1, the notion of robust AE (RAE), by Hoang, Krovetz and Rogaway [9], aims to capture a very strong security goal. The RAE security is captured as indistinguishability of a scheme from a particular idealized primitive in an unrestricted chosen ciphertext attack. The **rae** advantage of an adversary $\mathscr{A}$ against a scheme Π is defined as $\mathbf{Adv}^{\mathbf{rae}}_{\Pi}(\mathscr{A}) = \Pr[\mathscr{A}^{\mathbf{rae\text{-}R}_{\Pi}} \Rightarrow 1] - \Pr[\mathscr{A}^{\mathbf{rae\text{-}I}_{\Pi}} \Rightarrow 1]$ where the corresponding security games are defined in Figure 4.

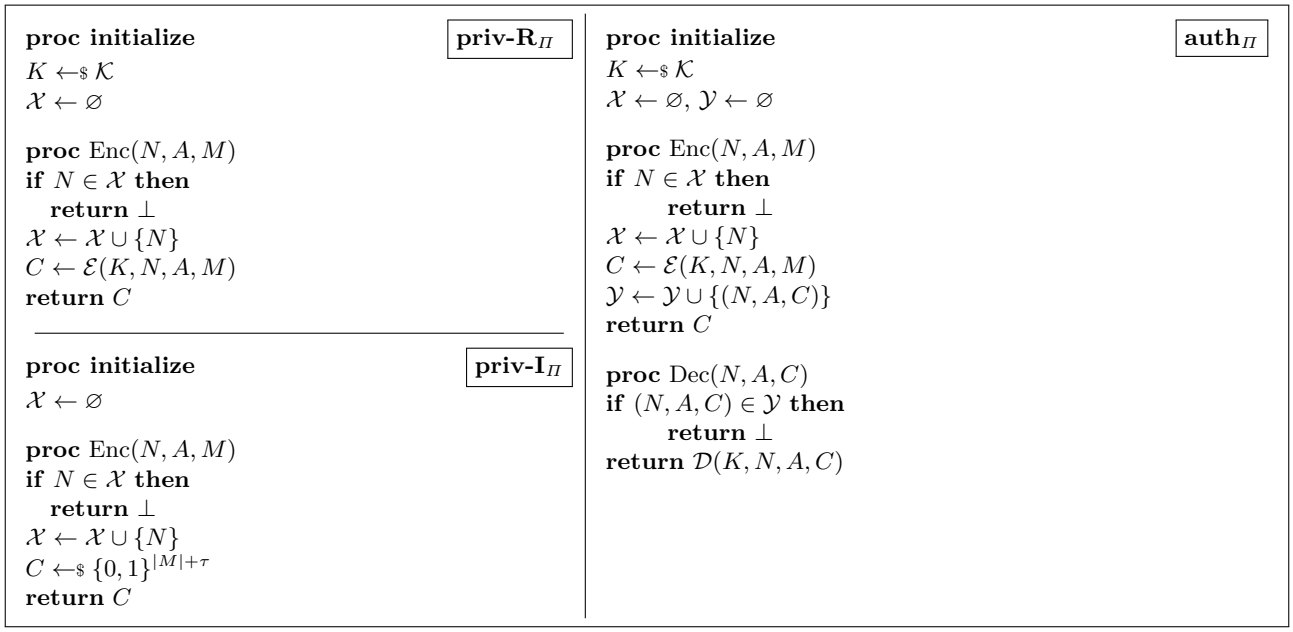

Fig. 2: **Two-requirement definition** of nAE security for a scheme $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ with ciphertext expansion τ.

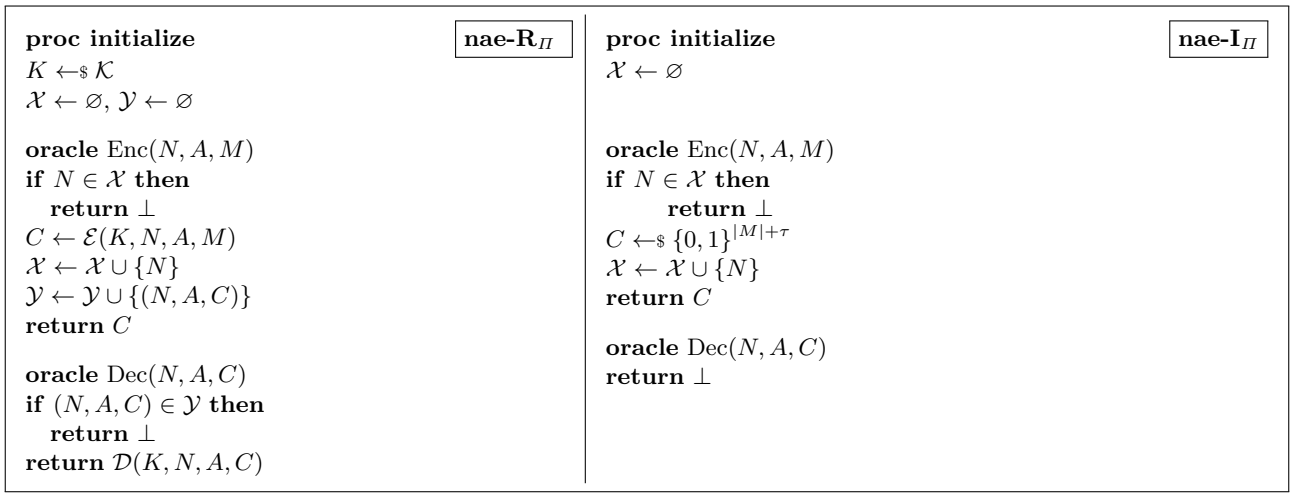

Fig. 3: **All-in-one definition** of nAE security for a scheme $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ with ciphertext expansion τ.

It is known that the strong RAE security of a scheme implies its nAE security. This can be easily verified by showing that $\mathbf{Adv}^{\mathbf{priv}}_{\Pi}(\mathscr{B}) \leq \mathbf{Adv}^{\mathbf{rae}}_{\Pi}(\mathscr{A})$ and $\mathbf{Adv}^{\mathbf{auth}}_{\Pi}(\mathscr{C}) \leq \mathbf{Adv}^{\mathbf{rae}}_{\Pi}(\mathscr{A}) + \frac{q_d}{2^\tau}$ for some adversaries $\mathscr{B}$ and $\mathscr{C}$ with the same resources as $\mathscr{A}$, q_d the number of decryption queries and τ the amount of stretch in all queries. However, the robustness of RAE comes at the expense of efficiency; an RAE-secure AE scheme must be inherently "offline", i.e. it cannot encrypt a plaintext with constant memory while outputting ciphertext bits with constant latency, as every bit of the ciphertext must depend on every bit of plaintex.

Stretch (in)dependent advantage. For some of the security notions we discuss, the adversarial advantage is trivially dependent on the value of stretch. The advantage for notions that capture integrity of ciphertexts will necessarily be high whenever stretch τ is low, as there is always a trivial attack that queries a random ciphertext with probability $2^{-\tau}$ of being successfully decrypted. This concerns the notions **auth** and **nae**. The notions that do not *directly* capture integrity of ciphertexts are not inherently impacted by the value of τ. In particular, no trivial attack with advantage $2^{-\tau}$ exists for the notions **priv** or **rae**. Note that **rae** captures the integrity property indirectly; the idealized reference of RAE security itself will still yield to the trivial attack mentioned above.

rae-R$_\Pi$

proc initialize
$K \leftarrow\$ \mathcal{K}$

proc Enc(N, A, τ, M)
return $\mathcal{E}(K, N, A, \tau, M)$

proc Dec(N, A, τ, C)
return $\mathcal{D}(K, N, A, \tau, C)$

rae-I$_\Pi$

proc initialize
for $N, A, \tau \in \mathcal{N} \times \{0,1\}^* \times \mathbb{N}$ **do** $\pi_{N,A,\tau} \leftarrow\$ \mathrm{Inj}(\tau)$

proc Enc(N, A, τ, M)
return $\pi_{N,A,\tau}(M)$

proc Dec(N, A, τ, C)
if $\exists M \in \{0,1\}^*$ s.t. $\pi_{N,A,\tau}(M) = C$ **then**
  return M
return $\perp$

Fig. 4: **RAE security.** Defining security for a robust AE scheme $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ with nonce space $\mathcal{N}$. $\mathrm{Inj}(\tau)$ denotes the set of all injective, τ-expanding functions from $\{0,1\}^*$ to $\{0,1\}^{\geq\tau}$.

3 Failure of Inserting Stretch into Nonce and/or AD

Using a generic forgery attack, we show that the recently proposed heuristic measures, namely, inclusion of the tag length in the nonce [21], in the AD [17] or in both nonce and AD fail when applied to a large class of nAE schemes (including e.g. GCM and OCB) that follow the "ciphertext translation" design paradigm of Rogaway [22] which is depicted in Figure 5. The attack is not completely new, it is a rather straightforward generalization of the tag-length misusing attack originally proposed by the Ascon team on a specific algorithm, namely OMD version 1 [6] which also follows the ciphertext translation method.

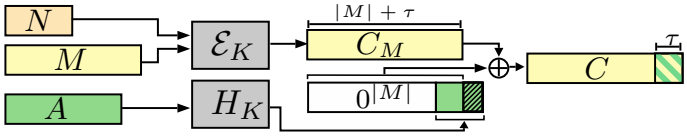

Fig. 5: **Ciphertext translation.** The message-only nAE encryption $\mathcal{E}$ produces an intermediate ciphertext C_M with τ bits of stretch. The leftmost τ bits of the output of a keyed hash $H_K(A)$ are are xored to the rightmost τ bits of C_M, forming the final ciphertext C.

The attack. We target a ciphertext translation-based AEAD scheme Π that supports any amount of stretch from a set $\mathcal{I}_T = \{\tau_1, \ldots, \tau_r\}$ with $\tau_1 < \tau_2 < \ldots < \tau_r$. We assume oracle access to encryption and decryption algorithms, such that the amount of stretch can be chosen for every query independently. The goal is to forge a ciphertext for A, M expanded by $\tau_g \in \mathcal{I}_T$ bits, with $g > 1$. The attack proceeds as in Figure 6. We let $\mathsf{left}_i(X)$ and $\mathsf{right}_j(X)$ denote i leftmost bits and j rightmost bits of a string X respectively.

1: $\Delta_A \leftarrow \varepsilon$; $A^* \leftarrow\$ \mathcal{A}\backslash\{A\}$
2: **for** $i \leftarrow 1$ **to** g **do**
3: pick fresh nonce N_i
4: $C_i^* \leftarrow \mathrm{Enc}(N_i, A^*, \tau_i, M)$
5: **do**
6: pick fresh $\delta \in \{0,1\}^{\tau_i - \tau_{i-1}}$
7: $C_i \leftarrow C_i^* \oplus 0^{|C_i| - \tau_i} \| \Delta_A \| \delta$
8: $M_i \leftarrow \mathrm{Dec}(N_i, A, \tau_i, C_i)$
9: **while** $M_i = \perp$
10: $\Delta_A \leftarrow \mathsf{right}_{\tau_i}(C_i \oplus C_i^*)$
11: **return** N_g, A, C_g

Fig. 6: **Ciphertext forgery** for a ciphertext translation-based AEAD scheme with associated data A and message M in presence of variable stretch. Here $\tau_0 = 0$.

The hash function $H_K(\cdot)$ used to process AD must fulfil some mild conditions for the attack to work against the heuristic countermeasures proposed in [17,21], namely:

- In case that the tag length is only injected into the nonce, the attack works with *arbitrary* $H_K(\cdot)$.
- For inclusion of the tag length in the AD or a combination of this method and nonce stealing, the attack works if $H_K(A) = H_{1_K}(A_1) \oplus H_{2_K}(A_2) \oplus \cdots \oplus H_{m_K}(A_m)$, for arbitrary functions H_{i_K} , $1 \leq i \leq m$, where $A = A_1 || A_2 || \cdots || A_m$ for $A_j \in \{0,1\}^n$ for some positive integer n (this is the case for both GCM and OCB). In this case, we must ensure that the block of AD that contains the amount of stretch τ is unchanged between A and A^*.

Under these conditions, the attack will always succeed: whenever we encrypt a message M with two different associated data A, A^*, first with τ_i and then with $\tau_j > \tau_i$ bits of stretch, then $C_i \oplus C_i^*$ will be a prefix of $C_j \oplus C_j^*$, as the xor cancels out the core ciphertext as well as the block of AD that is impacted by τ (if any).

The complexity of the attack in terms of verification queries will be $O(2^\mu)$ with $\mu = \max\{\tau_1, \tau_2 - \tau_1, \ldots, \tau_g - \tau_{g-1}\}$. For example, an adversary having access to the instances of the algorithm with 32-bit, 64-bit, 96-bit and 128-bit tags under the same key will only need a query complexity $O(2^{32})$ to forge a message with a 128-bit tag, which is in stark contrast with the expected $O(2^{128})$ query complexity.

4 Formalizing Nonce-based AE with Variable Stretch

Defining a meaningful security notion for AE schemes with variable stretch under the same key has turned out to be a non-trivial task [14, 21, 28]. Allowing the adversary to choose the amount of stretch freely from a set $\mathcal{I}_T = \{\tau_{\min}, \ldots, \tau_{\max}\}$ will inevitably enable it to produce forgeries with a high probability $2^{-\tau_{\min}}$ by targeting the shortest allowed stretch; a forgery is sure to be found with at most $2^{\tau_{\min}}$ verification queries. This is inherent to *any* AE scheme.

Despite this limit to its *global* security guarantees, there is a meaningful security property which *can* be expected from an nvAE scheme by a user: the scheme must guarantee τ bits of security for ciphertexts with τ bits of stretch, regardless of adversarial access to other instances with the same key but other (shorter and/or longer) amount of stretch than τ. For example, forging a ciphertext with τ-bit stretch should require $\approx 2^\tau$ verification queries *with τ-bit stretch*, regardless of the number of queries made under other different amounts of stretch.

This non-interference between different instances that use the same key but different stretch (tag length) is the intuition behind a formal definition for the notion of nonce-based, variable-stretch AE.

Security Definition. We define a security notion parameterized by the challenge stretch value $\tau_c \in \mathcal{I}_T$ as a natural extension to the notion of nAE. This is done in the compact all-in-one definition style of [26].

Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be a nvAE scheme whose syntax is defined in Section 2. An $\mathbf{nvae}(\tau_c)$ adversary $\mathscr{A}$ gets to interact with games $\mathbf{nvae}(\tau_c)\text{-}\mathbf{R}_\Pi$ (left) and $\mathbf{nvae}(\tau_c)\text{-}\mathbf{I}_\Pi$ (right) in Figure 7, defining respectively the real and ideal behavior of such a scheme. The adversary has access to two oracles Enc and Dec determined by these games and its goal is to distinguish the two games.

The adversary must respect a *relaxed nonce-requirement*; it must use a unique pair of nonce and stretch for encryption queries. Compared to the standard nonce-respecting requirement in nAE schemes, here nonce may be reused provided that the stretch does not repeat simultaneously.

In the ideal game $\mathbf{nvae}(\tau_c)\text{-}\mathbf{I}_\Pi$, the encryption and decryption queries with τ_c-bit stretch are answered in the same idealized way as in the "ideal" game of **nae** notion (Figure 3 right). However, the queries with stretch other than τ_c are treated with the real encryption/decryption algorithm. This lets the adversary to issue arbitrary queries (e.g. repeated forgeries) for any stretch $\tau \neq \tau_c$ and leverage the information thus gathered to attack the challenge expansion. At the same time, only queries with τ_c bits of stretch can help the adversary to actually distinguish the two games, capturing the exact level of security for queries with τ_c bits of stretch in presence of variable stretch.

We measure the advantage of $\mathscr{A}$ in breaking the $\mathbf{nvae}(\tau_c)$ security of Π as $\mathbf{Adv}_\Pi^{\mathbf{nvae}(\tau_c)}(\mathscr{A}) = \Pr[\mathscr{A}^{\mathbf{nvae}(\tau_c)\text{-}\mathbf{R}_\Pi} \Rightarrow 1] - \Pr[\mathscr{A}^{\mathbf{nvae}(\tau_c)\text{-}\mathbf{I}_\Pi} \Rightarrow 1]$.

Adversarial resources. The adversarial resources of interest for the $\mathbf{nvae}(\tau_c)$ notion are $(t, \mathbf{q_e}, \mathbf{q_d}, \boldsymbol{\sigma})$, 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 $\tau \in \mathcal{I}_T$, and $\mathbf{q_d} = (q_d^\tau | \tau \in \mathcal{I}_T)$ denotes the same for the decryption queries and $\boldsymbol{\sigma} = (\sigma^\tau | \tau \in \mathcal{I}_T)$ denotes the vector that holds the total amount of data σ^τ processed in all queries with stretch τ for every $\tau \in \mathcal{I}_T$.

Despite being focused on queries stretched by τ_c bits, we watch adversarial resources for every stretch $\tau \in \mathcal{I}_T$ in a detailed, vector-based fashion. This approach appears to be most flexible w.r.t. the security analysis. However, in a typical case we will be interested in the resources related to τ_c (i.e. $q_e^{\tau_c}, q_d^{\tau_c}, \sigma^{\tau_c}$) and cumulative resources of the adversary q_e, q_d, σ with $q_e = \sum_{\tau \in \mathcal{I}_T} q_e^\tau$, $q_d = \sum_{\tau \in \mathcal{I}_T} q_d^\tau$ and $\sigma = \sum_{\tau \in \mathcal{I}_T} \sigma^\tau$.

Remark 1 (Relation to nAE). The notion of $\mathbf{nvae}(\tau_c)$ is indeed an extension of the classical all-in-one security notion for nonce-based AE schemes. If the scheme Π is secure with some stretch-space $\mathcal{I}_T$, then it will be secure for any stretch-space $\mathcal{I}'_T \subseteq \mathcal{I}_T$, in particular for $\mathcal{I}'_T = \{\tau_c\}$. If a scheme has a stretch-space $\mathcal{I}_T = \{\tau_c\}$, then $\mathbf{nvae}(\tau_c)$ becomes the classical **nae** notion. It easily follows, that $\mathbf{nvae}(\tau_c)$ security of a scheme Π tightly implies **nae** security of $\Pi[\tau_c]$.

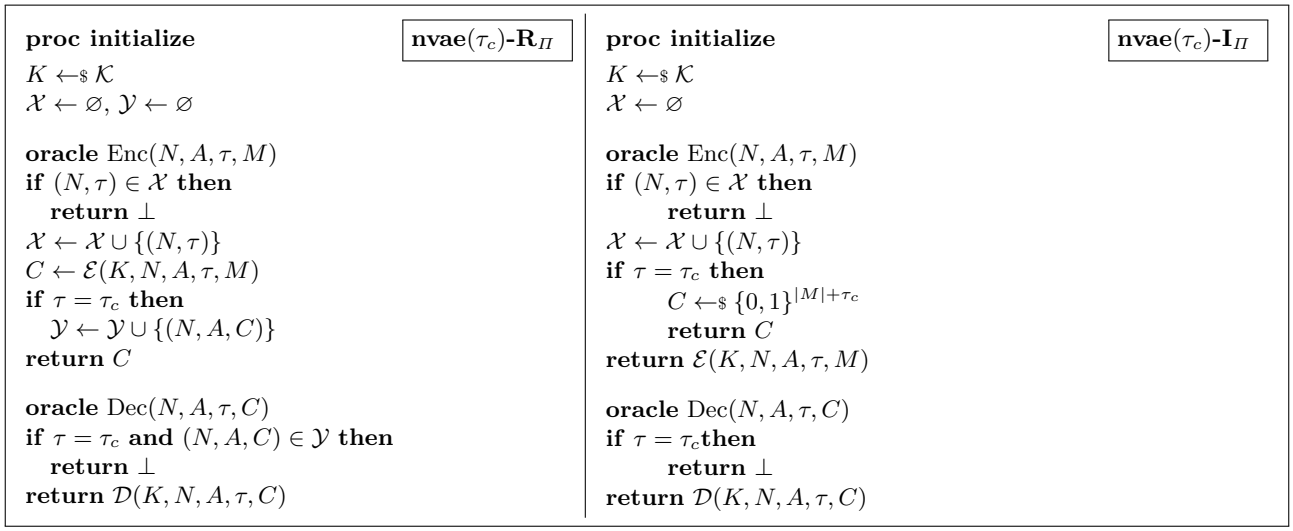

Fig. 7: **AE security with variable stretch.** Security games for defining AE security of a nonce-based AE scheme $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ with variable-stretch.

Similar to the **nae** notion, the **nvae**(τ_c) adversarial advantage will be trivially high if τ_c is low (due to successful forgeries). Yet, if the **nvae**(τ_c) advantage of a scheme behaves "reasonably", we will call the scheme secure. We discuss the interpretation of the **nvae**(τ_c) bounds in Appendix A.

Parameterized CCA security. An **nae**-secure AE scheme is also **ind-cca**-secure. This follows from the equivalence of the all-in-one and dual nAE notions and a well-known implication **priv** $\wedge$ **auth** $\Rightarrow$ **ind-cca** [2]. It is natural to ask: *Does the* ***nvae***(τ_c)*-security also provide a privacy guarantee against chosen ciphertext attacks?* We define a τ_c-parameterized extension of the **ind-cca** security notion and answer this question positively.

The parameterized **ind-cca**(τ_c) notion captures the exact privacy level guaranteed by an nvAE scheme for encryption queries stretched by τ_c bits, in presence of arbitrary queries with expansions $\tau \neq \tau_c$ and reasonable decryption queries stretched by τ_c bits. The notion is building on the intuition that privacy level of τ_c-expanded queries should not be affected by the adversarial queries with other amounts of stretch.

Security definition. Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be an nvAE with syntax defined in Section 2. We let an adversary $\mathscr{A}$ interact with the games **ind-cca**(τ_c)-$\mathbf{R}_{\Pi}$ and **ind-cca**(τ_c)-$\mathbf{I}_{\Pi}$ defined in Figure 8 and its goal is to distinguish them. In the "ideal" game **ind-cca**(τ_c)-$\mathbf{I}_{\Pi}$, the τ_c-stretched encryption queries are answered with random strings while the decryption queries are processed with the real decryption algorithm. $\mathscr{A}$ must respect the relaxed nonce-requirement and is prevented to win the game trivially (i.e. by re-encrypting output of decryption query with τ_c bits of stretch and vice-versa). We measure $\mathscr{A}$'s advantage in breaking **ind-cca**(τ_c) security of Π as $\mathbf{Adv}^{\mathbf{ind\text{-}cca}(\tau_c)}_{\Pi}(\mathscr{A}) = \Pr\left[\mathscr{A}^{\mathbf{ind\text{-}cca}(\tau_c)\text{-}\mathbf{R}} \Rightarrow 1\right] - \Pr\left[\mathscr{A}^{\mathbf{ind\text{-}cca}(\tau_c)\text{-}\mathbf{I}} \Rightarrow 1\right]$.
The adversarial resources of interest for the **ind-cca**(τ_c) notion are the same as for the **nvae**(τ_c) notion, i.e. $(t, \mathbf{q_e}, \mathbf{q_d}, \boldsymbol{\sigma})$.

Remark 2 (Relations to ind-cca and nvAE). Similarly as in the case of **nvae**(τ_c) and **nae**, **ind-cca**(τ_c) security with some stretch space $\mathcal{I}_T$ implies **ind-cca**(τ_c) security with any stretch space $\mathcal{I}'_T \subseteq \mathcal{I}_T$, e.g. $\mathcal{I}_T = \{\tau_c\}$. It follows that **ind-cca**(τ_c) security of a scheme Π implies the classical **ind-cca** security of $\Pi[\tau_c]$.

The notions of **ind-cca**(τ_c) and **nvae**(τ_c) differ mainly in the way the "ideal" games treat the decryption queries expanded by τ_c bits. The impact of this difference is substantial; the **ind-cca**(τ_c) notion does not capture integrity of ciphertexts. E.g. a scheme that concatenates output of a length-preserving, nonce-based, ind-cca-secure encryption scheme (using encoding of the nonce and stretch as a "nonce") and an image of the nonce and stretch under a PRF would be secure in the sense of **ind-cca**(τ_c), but insecure in the sense of **nvae**(τ_c).

We examine the relation between the two notions in the other direction in Theorem 1. We would like to stress that the result in Theorem 1 holds for *any* nvAE scheme, and in particular for any stretch space $\mathcal{I}_T$.

Theorem 1 (nvae$(\tau_c) \Rightarrow$ **ind-cca**(τ_c)**).** *Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be an arbitrary nonce-based AE scheme with variable stretch. We have that*

$$\mathbf{Adv}^{\boldsymbol{ind\text{-}cca}(\tau_c)}_{\Pi}(t, \mathbf{q_e}, \mathbf{q_d}, \boldsymbol{\sigma}) \leq 2 \cdot \mathbf{Adv}^{\boldsymbol{nvae}(\tau_c)}_{\Pi}(t', \mathbf{q_e}, \mathbf{q_d}, \boldsymbol{\sigma}),$$

ind-cca(τ_c)-$\mathbf{R}_{\Pi}$

proc initialize
$K \leftarrow\$ \mathcal{K}$
$\mathcal{V} \leftarrow \varnothing$, $\mathcal{X} \leftarrow \varnothing$, $\mathcal{Y} \leftarrow \varnothing$

oracle Enc(N, A, τ, M)
if $(N, \tau) \in \mathcal{X}$ **then return** $\perp$
if $\tau = \tau_c$ **and** $(N, A, M) \in \mathcal{V}$ **then**
  return $\perp$
$\mathcal{X} \leftarrow \mathcal{X} \cup \{(N, \tau)\}$
$C \leftarrow \mathcal{E}(K, N, A, \tau, M)$
if $\tau = \tau_c$ **then**
  $\mathcal{Y} \leftarrow \mathcal{Y} \cup \{(N, A, C)\}$
return C

oracle Dec(N, A, τ, C)
if $\tau = \tau_c$ **and** $(N, A, C) \in \mathcal{Y}$ **then**
  return $\perp$
$M \leftarrow \mathcal{D}(K, N, A, \tau, C)$
if $\tau = \tau_c$ **and** $M \neq \perp$
  $\mathcal{V} \leftarrow \mathcal{V} \cup \{(N, A, M)\}$
return M

ind-cca(τ_c)-$\mathbf{I}_{\Pi}$

proc initialize
$K \leftarrow\$ \mathcal{K}$
$\mathcal{V} \leftarrow \varnothing$, $\mathcal{X} \leftarrow \varnothing$, $\mathcal{Y} \leftarrow \varnothing$

oracle Enc(N, A, τ, M)
if $(N, \tau) \in \mathcal{X}$ **then return** $\perp$
if $\tau = \tau_c$ **and** $(N, A, M) \in \mathcal{V}$ **then**
    return $\perp$
$\mathcal{X} \leftarrow \mathcal{X} \cup \{(N, \tau)\}$
if $\tau = \tau_c$ **then**
    $C \leftarrow\$ \{0,1\}^{|M|+\tau_c}$
    $\mathcal{Y} \leftarrow \mathcal{Y} \cup \{(N, A, C)\}$
    return C
return $\mathcal{E}(K, N, A, \tau, M)$

oracle Dec(N, A, τ, C)
if $\tau = \tau_c$ **and** $(N, A, C) \in \mathcal{Y}$ **then**
    return $\perp$
$M \leftarrow \mathcal{D}(K, N, A, \tau, C)$
if $\tau = \tau_c$ **and** $M \neq \perp$
    $\mathcal{V} \leftarrow \mathcal{V} \cup \{(N, A, M)\}$
return M

Fig. 8: **Parameterized ind-cca security.** Games for defining **ind-cca**(τ_c) security of a nonce-based AE scheme with variable-stretch $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$.

with $t' = t + O(q)$ *and* $q = \sum_{\tau \in \mathcal{I}_{\mathcal{T}}} (q_e^{\tau} + q_d^{\tau})$.

Proof. Let $\mathscr{A}$ be an **ind-cca** adversary with indicated resources. We define the game **ind-cca**(τ_c)-$\mathbf{I}_{\Pi}^{\perp}$ as an intermediate step in the proof; it is exactly the same as **ind-cca**(τ_c)-$\mathbf{I}_{\Pi}$, except that the decryption queries with τ_c bits of stretch are always answered with $\perp$. We have that

$$\begin{aligned}\mathbf{Adv}_{\Pi}^{\mathbf{ind\text{-}cca}(\tau_c)}(\mathscr{A}) = \Pr[\mathscr{A}^{\mathbf{ind\text{-}cca}(\tau_c)\text{-}\mathbf{R}_{\Pi}} \Rightarrow 1] - \Pr[\mathscr{A}^{\mathbf{ind\text{-}cca}(\tau_c)\text{-}\mathbf{I}_{\Pi}^{\perp}} \Rightarrow 1] \\ + \Pr[\mathscr{A}^{\mathbf{ind\text{-}cca}(\tau_c)\text{-}\mathbf{I}_{\Pi}^{\perp}} \Rightarrow 1] - \Pr[\mathscr{A}^{\mathbf{ind\text{-}cca}(\tau_c)\text{-}\mathbf{I}_{\Pi}} \Rightarrow 1].\end{aligned}$$

We start by showing that $\Pr[\mathscr{A}^{\mathbf{ind\text{-}cca}(\tau_c)\text{-}\mathbf{R}_{\Pi}} \Rightarrow 1] - \Pr[\mathscr{A}^{\mathbf{ind\text{-}cca}(\tau_c)\text{-}\mathbf{I}_{\Pi}^{\perp}} \Rightarrow 1] \leq \mathbf{Adv}_{\Pi}^{\mathbf{nvae}(\tau_c)}(\mathscr{B})$ for an **nvae**(τ_c) adversary $\mathscr{B}$ with the resources $(t', \mathbf{q_e}, \mathbf{q_d}, \boldsymbol{\sigma})$. The reduction of $\mathscr{A}$ to $\mathscr{B}$ is straightforward: $\mathscr{B}$ simply answers $\mathscr{A}$'s queries with its own oracles, making sure that the trivial win-preventing restrictions of **ind-cca**(τ_c) games are met. At the end of experiment, $\mathscr{B}$ outputs whatever $\mathscr{A}$ outputs. This ensures perfect simulation of both games for $\mathscr{A}$.

It remains to show that $\Pr[\mathscr{A}^{\mathbf{ind\text{-}cca}(\tau_c)\text{-}\mathbf{I}_{\Pi}^{\perp}} \Rightarrow 1] - \Pr[\mathscr{A}^{\mathbf{ind\text{-}cca}(\tau_c)\text{-}\mathbf{I}_{\Pi}} \Rightarrow 1] \leq \mathbf{Adv}_{\Pi}^{\mathbf{nvae}(\tau_c)}(\mathscr{C})$ for an **nvae**(τ_c) adversary $\mathscr{C}$ with the resources $(t', \mathbf{q_e}, \mathbf{q_d}, \boldsymbol{\sigma})$. We reduce $\mathscr{A}$ to $\mathscr{C}$ as follows. $\mathscr{C}$ answers all $\mathscr{A}$'s queries directly with its own oracles (again making sure to enforce all the restrictions of **ind-cca**(τ_c) games), except for encryption queries expanded by τ_c bits. For those, $\mathscr{C}$ ignores its encryption oracle and answers with $|M| + \tau_c$ random bits if $\mathscr{A}$'s query has a fresh nonce-stretch pair an is not a re-encryption. At the end of experiment, $\mathscr{C}$ outputs the inverse of $\mathscr{A}$'s output. If $\mathscr{C}$ interacts with **nvae**(τ_c)-$\mathbf{R}_{\Pi}$, then it perfectly simulates **ind-cca**(τ_c)-$\mathbf{I}_{\Pi}$ for $\mathscr{A}$ while if $\mathscr{C}$ interacts with **nvae**(τ_c)-$\mathbf{I}_{\Pi}$, then it perfectly simulates **ind-cca**(τ_c)-$\mathbf{I}_{\Pi}^{\perp}$. □

No Two-Requirement Notion. The equivalence of the two-requirement (privacy and authenticity) approach and all-in-one approach for defining AE security is among the best known results in AE [26]. One may wonder whether such an equivalence also holds in the setting of variable-stretch AE schemes for natural τ_c-parameterized extensions of these notions. Surprisingly, we answer this question negatively. We consider the conventional privacy (ind-cpa$) and authenticity (integrity of ciphertexts) notions for AE schemes [2, 22] and define the notions of τ_c-privacy and τ_c-authenticity as natural parameterized extensions of their conventional counterparts.

Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be an nvAE scheme with syntax defined in Section 2. An adversary $\mathscr{A}$ against τ_c-privacy of Π interacts with games **priv**(τ_c)-$\mathbf{R}_{\Pi}$ (real scheme) and **priv**(τ_c)-$\mathbf{I}_{\Pi}$ (ideal behaviour) defined in Figure 9, and tries to distinguish them. We measure $\mathscr{A}$'s advantage in breaking the τ_c-privacy of Π in a chosen plaintext attack as $\mathbf{Adv}_{\Pi}^{\mathbf{priv}(\tau_c)}(\mathscr{A}) = \Pr[\mathscr{A}^{\mathbf{priv}(\tau_c)\text{-}\mathbf{R}_{\Pi}} \Rightarrow 1] - \Pr[\mathscr{A}^{\mathbf{priv}(\tau_c)\text{-}\mathbf{I}_{\Pi}} \Rightarrow 1]$.

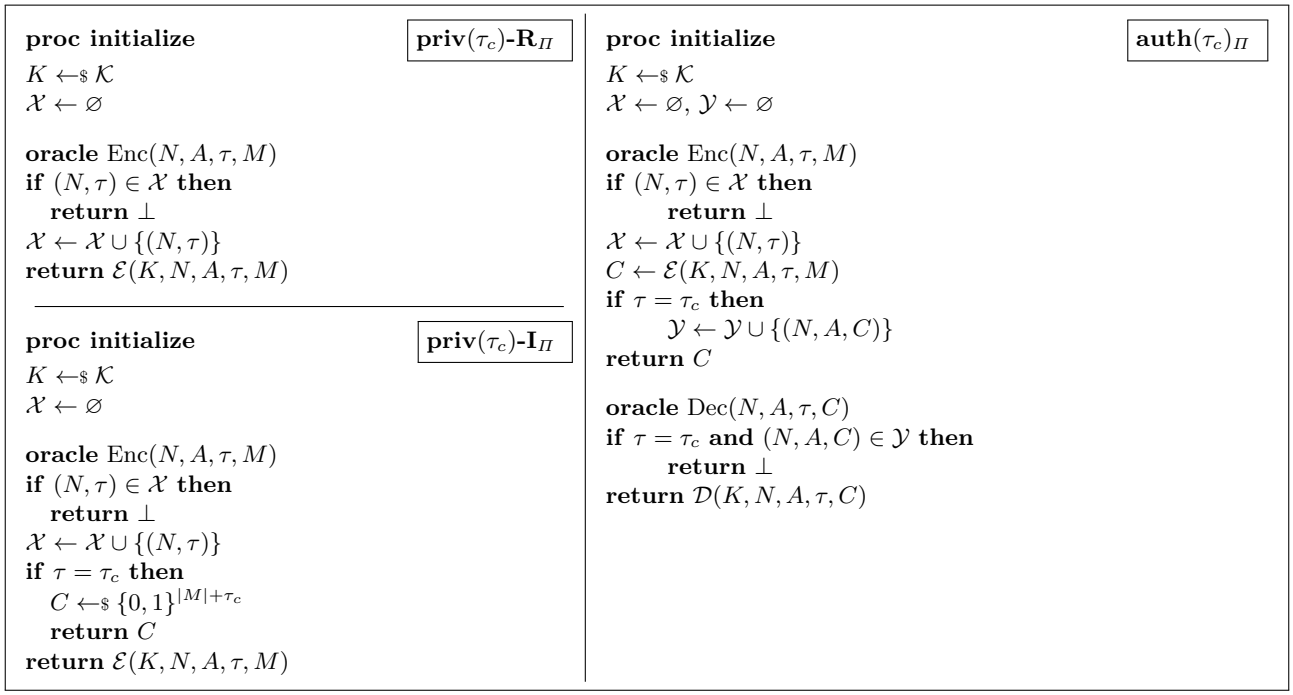

Fig. 9: **Dual nvAE security.** Security games for defining AE security of a nonce-based AE scheme $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ with variable-stretch.

An adversary $\mathscr{A}$ that attacks the τ_c-authenticity of Π is left to interact with the game $\mathbf{auth}(\tau_c)_\Pi$ defined in Figure 9 and its goal is to find a valid forgery (i.e. produce a decryption query returning $M \neq \perp$) with the target stretch of τ_c bits. We measure the advantage of $\mathscr{A}$ in breaking τ_c-authenticity of Π in a chosen ciphertext attack by $\mathbf{Adv}_{\Pi}^{\mathbf{auth}(\tau_c)}(\mathscr{A}) = \Pr\left[\mathscr{A}^{\mathbf{auth}(\tau_c)_\Pi} \text{ forges with } \tau_c\right]$.

The adversarial resources of interest for the $\mathbf{priv}(\tau_c)$ and $\mathbf{auth}(\tau_c)$ notions are $(t, \mathbf{q_e}, \boldsymbol{\sigma})$ and $(t, \mathbf{q_e}, \mathbf{q_d}, \boldsymbol{\sigma})$ respectively, defined as for the notion of $\mathbf{nvae}(\tau_c)$ in the current Section.

Remark 3 (Relations with the all-in-one nvAE, priv and auth notions). As before, if a scheme Π is $\mathbf{priv}(\tau_c)$ ($\mathbf{auth}(\tau_c)$) secure with stretch-space $\mathcal{I}_T$, then it will be secure for any stretch-space $\mathcal{I}'_T \subseteq \mathcal{I}_T$ including $\mathcal{I}'_T = \{\tau_c\}$, implying the **priv** (**auth**) security of the scheme $\Pi[\tau_c]$.

We can easily verify that the $\mathbf{nvae}(\tau_c)$ security of a scheme Π implies both the $\mathbf{priv}(\tau_c)$ security and the $\mathbf{auth}(\tau_c)$ of Π, by adapting the reductions for corresponding conventional notions [26] slightly. In Proposition 1, we show that the converse of this implication does not hold.

Proposition 1. *There exists a nonce-based AE scheme with variable stretch, that is secure in the sense of both the* $\boldsymbol{priv}(\tau_c)$ *notion and the* $\boldsymbol{auth}(\tau_c)$ *notion but insecure in the sense of* $\boldsymbol{ind\text{-}cca}(\tau_c)$ *notion, i.e.*

$$\boldsymbol{priv}(\tau_c) \wedge \boldsymbol{auth}(\tau_c) \not\Rightarrow \boldsymbol{ind\text{-}cca}(\tau_c),$$

assuming the existence of secure tweakable blockciphers and PRFs.

To support the claim in Proposition 1, we define the nvAE scheme $\Pi_{\neg\text{cca}} = (\mathcal{K}_{\neg\text{cca}}, \mathcal{E}_{\neg\text{cca}}, \mathcal{E}_{\neg\text{cca}})$ constructed from an ind-cpa secure tweakable blockcipher $\mathsf{B} : \mathcal{K}_1 \times \mathcal{N} \times \{0,1\}^n \to \{0,1\}^n$ and two PRFs $F : \mathcal{K}_2 \times \{0,1\}^* \to \{0,1\}^n$ and $F' : \mathcal{K}_3 \times \{0,1\}^* \to \{0,1\}^m$. We define $\mathcal{K}_{\neg\text{cca}} = \mathcal{K}_1 \times \mathcal{K}_2 \times \mathcal{K}_3$, $\mathcal{M}_{\neg\text{cca}} = \{0,1\}^n$, $\mathcal{A}_{\neg\text{cca}} = \{0,1\}^*$, $\mathcal{N}_{\neg\text{cca}} = \mathcal{N}$ and the encryption and decryption algorithms as in Figure 11. We require that $|\mathcal{I}_{T\,\neg\text{cca}}| \geq 2$ and that $m \geq \max(\mathcal{I}_{T\,\neg\text{cca}})$. The encryption algorithm $\mathcal{E}_{\neg\text{cca}}$ is depicted in Figure 10.

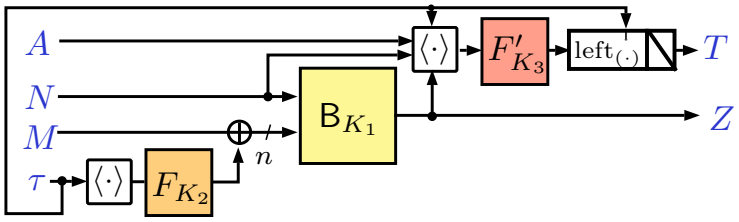

Fig. 10: The encryption algorithm of the scheme $\Pi_{\neg\text{cca}}$. $\langle\cdot\rangle$ is an efficiently computable, injective encoding scheme.

proc $\mathcal{E}_{\neg\text{cca}}(K, N, A, \tau, M)$ Parse K as K_1, K_2, K_3 $W \leftarrow M \oplus F(K_2, \langle\tau\rangle)$ $Z \leftarrow \mathsf{B}(K_1, N, W)$ $T \leftarrow \mathsf{left}_\tau(F'(K_3, \langle N, A, \tau, Z\rangle))$ **return** $Z\|T$	**proc** $\mathcal{D}_{\neg\text{cca}}(K, N, A, \tau, C)$ Parse K as K_1, K_2, K_3 Parse C as $Z\|T$ with $\|T\| = \tau$ **if** $\mathsf{left}_\tau(F'(K_3, \langle N, A, \tau, Z\rangle)) \neq T$ **then** **return** $\perp$ $W \leftarrow \mathsf{B}^{-1}(K_1, N, Z)$ **return** $W \oplus F(K_2, \langle\tau\rangle)$

Fig. 11: Encryption and decryption algorithms of the nonce-based, variable-stretch AE scheme $\Pi_{\neg\text{cca}} = (\mathcal{K}_{\neg\text{cca}}, \mathcal{E}_{\neg\text{cca}}, \mathcal{E}_{\neg\text{cca}})$. $\langle\cdot\rangle$ is an efficiently computable, injective encoding scheme.

The scheme $\Pi_{\neg\text{cca}}$ is by far no real-life AE construction (mainly due to its limited message space), its purpose is merely to act as a counter example. It can be verified, that $\mathbf{Adv}^{\mathbf{auth}(\tau_c)}_{\Pi_{\neg\text{cca}}}(t, \mathbf{q_e}, \mathbf{q_d}, \boldsymbol{\sigma}) \leq \mathbf{Adv}^{PRF}_{F'}(t, q_e + q_d, \sigma) + q_d^{\tau_c}/2^{\tau_c}$; every forgery attempt equals to guessing τ_c bits of an output of F', evaluated on a fresh input.[3] For privacy, we have that $\mathbf{Adv}^{\mathbf{priv}(\tau_c)}_{\Pi_{\neg\text{cca}}}(t, \mathbf{q_e}, \mathbf{q_d}, \boldsymbol{\sigma}) \leq \mathbf{Adv}^{PRF}_{F}(t, q_e, \sigma) + \mathbf{Adv}^{PRF}_{F'}(t, q_e, \sigma) + \mathbf{Adv}^{\widetilde{prp}}_{\mathsf{B}}(t, q_e) + 2q_e^2/2^n$. Here $q_e = \sum_{\tau\in\mathcal{I}_T} q_e^\tau$, $q_d = \sum_{\tau\in\mathcal{I}_T} q_d^\tau$ and $\sigma = \sum_{\tau\in\mathcal{I}_T} \sigma^\tau$.

The term $2q_e^2/2^n$ is composed of $q_e^2/2^n$ that comes from a *RP-RF* switch for the tweakable blockcipher and another $q_e^2/2^n$ that comes from extending the tweakspace to include stretch, using F (similar to the XE construction of [23]). However, we can construct an adversary $\mathscr{A}_{\neg\text{cca}}$, that achieves $\mathbf{ind\text{-}cca}(\tau_c)$ advantage close to 1. The strategy of $\mathscr{A}_{\neg\text{cca}}$ is as follows:

1. ask query $Z_1\|T_1 \leftarrow \text{Enc}(N_1, A_1, \tau_c, M_1)$ with arbitrary N_1, A_1, M_1,
2. iterate through $T_1^* \in \{0,1\}^{\tau_{\min}}$ until $M_1^* \leftarrow \text{Dec}(N_1, A_1, \tau_{\min}, Z_1\|T_1^*)$ returns $M_1^* \neq \perp$,
3. ask query $Z_2\|T_2 \leftarrow \text{Enc}(N_2, A_2, \tau_c, M_2)$ with arbitrary N_2, A_2, M_2,
4. iterate through $T_2^* \in \{0,1\}^{\tau_{\min}}$ until $M_2^* \leftarrow \text{Dec}(N_2, A_2, \tau_{\min}, Z_2\|T_2^*)$ returns $M_2^* \neq \perp$,
5. return 1 iff $M_1 \oplus M_1^* = M_2 \oplus M_2^*$ (otherwise return 0),

where $\tau_{\min} = \min(\mathcal{I}_T \backslash \{\tau_c\})$. We have that $\mathbf{Adv}^{\mathbf{ind\text{-}cca}(\tau_c)}_{\Pi_{\neg\text{cca}}}(\mathscr{A}_{\neg\text{cca}}) = 1 - 2^{-n}$. As amount of stretch τ has no effect on the encryption by B, we can verify that

$$\begin{aligned} M_1 \oplus F(K_2, \langle\tau_c\rangle) &= M_1^* \oplus F(K_2, \langle\tau_{\min}\rangle) \\ M_2 \oplus F(K_2, \langle\tau_c\rangle) &= M_2^* \oplus F(K_2, \langle\tau_{\min}\rangle) \end{aligned}$$

The final conditional statement verified by the adversary is always true for the real scheme. The probability of the same event in the "ideal" game is 2^{-n}. As a consequence of Theorem 1 and Proposition 1, we can state Corollary 1.[4]

Corollary 1. *There exists a nonce-based AE scheme with variable stretch, that is secure in the sense of both the* $\boldsymbol{priv}(\tau_c)$ *notion and the* $\boldsymbol{auth}(\tau_c)$ *notion but insecure in the sense of* $\boldsymbol{nvae}(\tau_c)$ *notion, i.e.*

$$\boldsymbol{priv}(\tau_c) \wedge \boldsymbol{auth}(\tau_c) \nRightarrow \boldsymbol{nvae}(\tau_c)$$

KEY-EQUIVALENT SEPARATION BY STRETCH. The notion of $\mathbf{nvae}(\tau_c)$ captures the immediate intuition about the security goal one expects to achieve using a nonce-based AE scheme with variable stretch. We now introduce a

[3] Note that τ_c is an index rather than a power in $q_d^{\tau_c}$.

[4] The same attack strategy yields also $\mathbf{Adv}^{\mathbf{nvae}(\tau_c)}_{\Pi_{\neg\text{cca}}}(\mathscr{A}_{\neg\text{cca}}) = 1 - 2^{-n}$.

kess-R$_\Pi$

proc initialize
$K \leftarrow\!\!\$\ \mathcal{K}$
$\mathcal{X} \leftarrow \varnothing$

oracle Enc(N, A, τ, M)
if $(N, \tau) \in \mathcal{X}$ **then**
  return $\perp$
$\mathcal{X} \leftarrow \mathcal{X} \cup \{(N, \tau)\}$
return $\mathcal{E}(K, N, A, \tau, M)$

oracle Dec(N, A, τ, C)
return $\mathcal{D}(K, N, A, \tau, C)$

kess-I$_A$E

proc initialize
for $\tau \in \mathcal{I}_T$ **do**
  $K_\tau \leftarrow\!\!\$\ \mathcal{K}$

oracle Enc(N, A, τ, M)
if $(N, \tau) \in \mathcal{X}$ **then**
  return $\perp$
$\mathcal{X} \leftarrow \mathcal{X} \cup \{(N, \tau)\}$
return $\mathcal{E}(K_\tau, N, A, \tau, M)$

oracle Dec(N, A, τ, C)
return $\mathcal{D}(K_\tau, N, A, \tau, C)$

Fig. 12: **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**$_A E$ can be done by lazy sampling if needed.

proc initialize
for $\tau \in \mathcal{I}_T$ **do**
  $K_\tau \leftarrow\!\!\$\ \mathcal{K}$
$\mathcal{X} \leftarrow \varnothing$, $\mathcal{Y} \leftarrow \varnothing$

oracle Dec(N, A, τ, C)
if $\tau = \tau_c$ **and** $(N, A, C) \in \mathcal{Y}$ **then**
  return $\perp$
return $\mathcal{D}(K_\tau, N, A, \tau, C)$

oracle Enc(N, A, τ, M)
if $(N, \tau) \in \mathcal{X}$ **then**
  return $\perp$
$\mathcal{X} \leftarrow \mathcal{X} \cup \{(N, \tau)\}$
$C \leftarrow \mathcal{E}(K_\tau, N, A, \tau, M)$
if $\tau = \tau_c$ **then**
  $\mathcal{Y} \leftarrow \mathcal{Y} \cup \{(N, A, C)\}$
return C

Fig. 13: Security game **nvae**(τ_c)-G_Π.

modular approach to *achieving* the notion. Assume that an AE scheme is already known to be secure in the sense of the nAE model. What additional security property should such a scheme possess (i.e. on top of nAE-security) so that it can achieve the full aim of being a **nvae**(τ_c)-secure scheme? We formalize such a desirable property, naming it *key-equivalent separation by stretch* (**kess**), which captures the intuition that for each value of stretch the scheme should behave as if keyed with a fresh, independent secret key.

Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be an nvAE scheme with the syntax defined in Section 2. We let an adversary $\mathscr{A}$ that tries to break **kess** of Π interact with games defined in Figure 12. The goal of the adversary is to distinguish these the two games. The advantage of $\mathscr{A}$ in breaking the **kess** property of the scheme Π is measured by $\mathbf{Adv}^{\mathbf{kess}}_{\Pi}(\mathscr{A}) = \Pr\left[\mathscr{A}^{\mathbf{kess\text{-}R}_A E} \Rightarrow 1\right] - \Pr\left[\mathscr{A}^{\mathbf{kess\text{-}I}_A E)} \Rightarrow 1\right]$.

The adversarial resources of interest for the 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.

We note that **kess** on its own says nothing about AE security of a scheme (e.g. identity "encryption" concatenated with τ zeroes achieves **kess**, but is far from **nae**-secure). However, we show in Theorem 2 that when combined with **nae** security, **kess** implies **nvae**(τ_c) security. Informally, the **kess** notion takes care of interaction between queries with different values of stretch. Once this is done, we are free to argue that the queries with τ_c bits of stretch are "independent" of those with other values of stretch and will "inherit" the security level of $\Pi[\tau_c]$.

Theorem 2 (kess $\wedge$ nae $\Rightarrow$ nvae(τ_c)). *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 $\sigma = \sum_{\tau \in \mathcal{I}_T}(\sigma_e^\tau + \sigma_d^\tau)$.

Proof. Let $\mathscr{A}$ be an **nvae**(τ_c) adversary with the indicated resources. Consider the security game **nvae**(τ_c)-G defined in Figure 13. We have that

$$\begin{aligned}\mathbf{Adv}^{\mathbf{nvae}(\tau_c)}_{\Pi}(\mathscr{A}) = {} & \Pr[\mathscr{A}^{\mathbf{nvae}(\tau_c)\text{-}\mathbf{R}_\Pi} \Rightarrow 1] - \Pr[\mathscr{A}^{\mathbf{nvae}(\tau_c)\text{-}G_\Pi} \Rightarrow 1] \\ & + \Pr[\mathscr{A}^{\mathbf{nvae}(\tau_c)\text{-}G_\Pi} \Rightarrow 1] - \Pr[\mathscr{A}^{\mathbf{nvae\text{-}I}_\Pi(\tau_c)} \Rightarrow 1].\end{aligned}$$

We first show that $\Pr[\mathscr{A}^{\mathbf{nvae}(\tau_c)\text{-}\mathbf{R}_\Pi} \Rightarrow 1] - \Pr[\mathscr{A}^{\mathbf{nvae}(\tau_c)\text{-}G_\Pi} \Rightarrow 1] \leq \mathbf{Adv}^{\mathbf{kess}}_{\Pi}(\mathscr{B})$ for a **kess** adversary $\mathscr{B}$ with the resources $(t', \mathbf{q_e}, \mathbf{q_d}, \boldsymbol{\sigma})$. The **nvae**$(\tau_c)$ adversary $\mathscr{A}$ can be straightforwardly reduced to $\mathscr{B}$. Any query of $\mathscr{A}$

```
101: Algorithm E_K(N, A, M)
102:     if N ∉ 𝒩 then
103:         return ⊥
104:     M_1‖M_2 ··· M_m‖M_* ← M where
105:         each |M_i| = n and |M_*| < n
106:     Sum ← 0^n, C_* ← ε
107:     for i ← 1 to m do
108:         C_i ← Ẽ_K^{N,i,0}(M_i)
109:         Sum ← Sum ⊕ M_i
110:     if M_* = ε then
111:         Final ← Ẽ_K^{N,m,2}(Sum)
112:     else
113:         Pad ← Ẽ_K^{N,m,1}(0^n)
114:         C_* ← M_* ⊕ left_{|M_*|}(Pad)
115:         Sum ← Sum ⊕ M_*‖10*
116:         Final ← Ẽ_K^{N,m,3}(Sum)
117:     Auth ← Hash_K(A)
118:     T ← left_τ(Final ⊕ Auth)
119:     return C_1‖C_2‖ ··· ‖C_m‖C_*‖T

301: Algorithm HASH_K(A)
302:     Sum ← 0^n
303:     A_1‖A_2 ··· A_m‖A_* ← A where
304:         each |A_i| = n and |A_*| < n
305:     for i ← 1 to m do
306:         Sum ← Sum ⊕ Ẽ_K^{i,0}(A_i)
307:     if A_* ≠ ε then
308:         Sum ← Sum ⊕ Ẽ_K^{m,1}(A_*‖10*)
309:     return Sum

201: Algorithm D_K(N, A, C)
202:     if N ∉ 𝒩 or |C| < τ then
203:         return ⊥
204:     C_1‖C_2 ··· C_m‖C_*‖T ← C where
205:         each where each |C_i| = n,
206:         |C_*| < n and |T| = τ
207:     Sum ← 0^n, M_* ← ε
208:     for i ← 1 to m do
209:         M_i ← D̃_K^{N,τ,i,0}(C_i)
210:         Sum ← Sum ⊕ M_i
211:     if C_* = ε then
212:         Final ← Ẽ_K^{N,m,2}(Sum)
213:     else
214:         Pad ← Ẽ_K^{N,m,1}(0^n)
215:         M_* ← C_* ⊕ left_{|C_*|}(Pad)
216:         Sum ← Sum ⊕ M_*‖10*
217:         Final ← Ẽ_K^{N,m,3}(Sum)
218:     Auth ← Hash_K(A)
219:     T' ← left_τ(Final ⊕ Auth)
220:     if T = T' then
221:         return C_1‖ ··· ‖C_m‖C_*‖T
222:     else
223:         return ⊥
```

Fig. 14: Definition of $\Theta\mathrm{CB}[\widetilde{E}, \tau]$.

is directly answered with $\mathscr{B}$'s own oracles, except for decryption queries with expansion of τ_c bits whose output is trivially known from previous encryption queries; here $\mathscr{B}$ returns $\perp$ to $\mathscr{A}$. At the end, $\mathscr{B}$ outputs whatever $\mathscr{A}$ outputs. If $\mathscr{B}$ interacts with **kess-R**$_{\Pi}$ then it perfectly simulates **nvae**(τ_c)-**R**$_{\Pi}$ for $\mathscr{A}$. If $\mathscr{B}$ interacts with **kess-I**$_{\Pi}$ then it perfectly simulates **nvae**(τ_c)-G_{Π}.
We next show that $\Pr[\mathscr{A}^{\mathbf{nvae}(\tau_c)\text{-}G_{\Pi}} \Rightarrow 1] - \Pr[\mathscr{A}^{\mathbf{nvae}\text{-}\mathbf{I}_{\Pi}(\tau_c)} \Rightarrow 1] \leq \mathbf{Adv}^{\mathbf{nae}}_{\Pi[\tau_c]}(\mathscr{C})$ for an **nae** adversary $\mathscr{C}$ with resources $(t'', q_e^{\tau_c}, q_d^{\tau_c}, \sigma^{\tau_c})$. $\mathscr{A}$ can be reduced to $\mathscr{C}$ in the following way. When $\mathscr{A}$ issues a query with expansion τ_c, $\mathscr{C}$ answers it with its own oracles. For other amounts of stretch $\tau \neq \tau_c$, $\mathscr{C}$ first checks if there were previous queries with τ bits of stretch. If not, it samples a fresh key K_τ. $\mathscr{C}$ then processes the query with the real (encryption or decryption) algorithm of Π and the key K_τ, making sure that encryption queries comply with the nonce requirement and are not re-encryptions. If $\mathscr{C}$ interacts with **nae-R**$_{\Pi[\tau_c]}$ then it perfectly simulates **nvae**(τ_c)-G_{Π} for $\mathscr{A}$. If $\mathscr{C}$ interacts with **nae-I**$_{\Pi[\tau_c]}$ then it perfectly simulates **nvae**(τ_c)-**I**$_{\Pi}$. This yields the desired result. □

Remark 4. An RAE secure scheme Π will always have the **kess** property. To see why, note that replacing Π by a collection of random injections in both the **kess-R**$_A E$ and **kess-I**$_A E$ games will not increase the advantage significantly, as that would contradict Π's RAE security. After the replacement, the two games will be indistinguishable.

5 Achieving AE with Variable Stretch

We demonstrate that the security of AE schemes in the sense of **nvae**(τ_c) notion is easily achievable by introducing a practical and secure scheme. Rather than constructing a scheme from the scratch, we modify an existing, well-established scheme and follow a modular approach to analyse its security in presence of variable stretch. The modification we propose is general enough to be applicable to most of the AE schemes based on a tweakable primitive (e.g. tweakable blockcipher).

OCB MODE FOR TWEAKABLE BLOCKCIPHER. The Offset Codebook mode of operation for a tweakable blockcipher (ΘCB) is a nonce-based AE scheme proposed by Krovetz and Rogaway [13] (prior versions of OCB were defined in [23, 25]). It is parameterized by a tweakable blockcipher $\widetilde{E} : \mathcal{K} \times \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$ and a tag length $0 \leq \tau \leq n$. The tweak space of $\widetilde{E}$ is of the form $\mathcal{T} = \mathcal{N} \times \mathbb{N}_0 \times \{0,1,2,3\} \cup \mathbb{N}_0 \times \{0,1,2,3\}$ for a finite set $\mathcal{N}$. The encryption and the decryption algorithms of $\Theta\mathrm{CB}[\widetilde{E}, \tau]$ are described in Figure 14. The security of ΘCB is captured in Lemma 1.

```
101: Algorithm $\mathcal{E}_K(N, A, \tau, M)$
102:   if $N \notin \mathcal{N}$ then
103:     return $\perp$
104:   $M_1 \| M_2 \cdots M_m \| M_* \leftarrow M$ where
105:       each $|M_i| = n$ and $|M_*| < n$
106:   Sum $\leftarrow 0^n$, $C_* \leftarrow \varepsilon$
107:   for $i \leftarrow 1$ to $m$ do
108:     $C_i \leftarrow \widetilde{E}_K^{N,\tau,i,0}(M_i)$
109:     Sum $\leftarrow$ Sum $\oplus\, M_i$
110:   if $M_* = \varepsilon$ then
111:     Final $\leftarrow \widetilde{E}_K^{N,\tau,m,2}$(Sum)
112:   else
113:     Pad $\leftarrow \widetilde{E}_K^{N,\tau,m,1}(0^n)$
114:     $C_* \leftarrow M_* \oplus \mathsf{left}_{|M_*|}$(Pad)
115:     Sum $\leftarrow$ Sum $\oplus\, M_* \| 10^*$
116:     Final $\leftarrow \widetilde{\mathrm{E}}_{\mathrm{K}}^{\mathrm{N},\tau,\mathrm{m},3}$(Sum)
117:   Auth $\leftarrow \mathrm{Hash}_K(A)$
118:   $T \leftarrow \mathsf{left}_\tau$(Final $\oplus$ Auth)
119:   return $C_1 \| C_2 \| \cdots \| C_m \| C_* \| T$

301: Algorithm $\mathrm{HASH}_K(A, \tau)$
302:   Sum $\leftarrow 0^n$
303:   $A_1 \| A_2 \cdots A_m \| A_* \leftarrow A$ where
304:       each s$|A_i| = n$ and $|A_*| < n$
305:   for $i \leftarrow 1$ to $m$ do
306:     Sum $\leftarrow$ Sum $\oplus\, \widetilde{E}_K^{\tau,i,0}(A_i)$
307:   if $A_* \neq \varepsilon$ then
308:     Sum $\leftarrow$ Sum $\oplus\, \widetilde{E}_K^{\tau,m,1}(A_* \| 10^*)$
309:   return Sum

201: Algorithm $\mathcal{D}_K(N, A, \tau, \mathsf{C})$
202:   if $N \notin \mathcal{N}$ or $|\mathsf{C}| < \tau$ then
203:     return $\perp$
204:   $C_1 \| C_2 \cdots C_m \| C_* \| T \leftarrow \mathsf{C}$ where
205:       each $|C_i| = n$,
206:       $|C_*| < n$ and $|T| = \tau$
207:   Sum $\leftarrow 0^n$, $M_* \leftarrow \varepsilon$
208:   for $i \leftarrow 1$ to $m$ do
209:     $M_i \leftarrow D_K^{N,\tau,i,0}(C_i)$
210:     Sum $\leftarrow$ Sum $\oplus\, M_i$
211:   if $C_* = \varepsilon$ then
212:     Final $\leftarrow \widetilde{E}_K^{N,\tau,m,2}$(Sum)
213:   else
214:     Pad $\leftarrow \widetilde{E}_K^{N,\tau,m,1}(0^n)$
215:     $M_* \leftarrow C_* \oplus \mathsf{left}_{|C_*|}$(Pad)
216:     Sum $\leftarrow$ Sum $\oplus\, M_* \| 10^*$
217:     Final $\leftarrow \widetilde{\mathrm{E}}_{\mathrm{K}}^{\mathrm{N},\tau,\mathrm{m},3}$(Sum)
218:   Auth $\leftarrow \mathrm{Hash}_K(A)$
219:   $T' \leftarrow \mathsf{left}_\tau$(Final $\oplus$ Auth)
220:   if $T = T'$ then
221:     return $C_1 \| \cdots \| C_m \| C_* \| T$
222:   else
223:     return $\perp$
```

Fig. 15: Definition of $\Theta\mathrm{CBv}[\widetilde{E}]$. Changes from ΘCB highlighted in red.

Lemma 1. *(Lemma 2 [13]) Let $\widetilde{E} : \mathcal{K} \times \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$ be a tweakable blockcipher with $\mathcal{T} = \mathcal{N} \times \mathbb{N}_0 \times \{0,1,2,3\} \cup \mathbb{N}_0 \times \{0,1,2,3\}$. Let $\tau \in \{0,\ldots,n\}$. Then we have that*

$$\mathbf{Adv}^{\boldsymbol{priv}}_{\Theta\mathrm{CB}[\widetilde{E},\tau]}(t, q_e, \sigma) \le \mathbf{Adv}^{\pm\widetilde{\mathrm{prp}}}_{\widetilde{E}}(t', q^p),$$

$$\mathbf{Adv}^{\boldsymbol{auth}}_{\Theta\mathrm{CB}[\widetilde{E},\tau]}(t, q_e, q_d, \sigma) \le \mathbf{Adv}^{\pm\widetilde{\mathrm{prp}}}_{\widetilde{E}}(t', q^a) + q_d \cdot \frac{2^{n-\tau}}{2^n - 1},$$

where $q^p \le \lceil \sigma/n \rceil + 2 \cdot q_e$, and $q^a \le \lceil \sigma/n \rceil + 2 \cdot (q_e + q_d)$, and $t' = t + O(\sigma)$.

Thanks to the results of [26, 27], we can state as a corollary of Lemma 1 that $\mathbf{Adv}^{\mathbf{nae}}_{\Theta\mathrm{CB}[\widetilde{E},\tau]}(t, q_e, q_d, \sigma) \le \mathbf{Adv}^{\pm\widetilde{\mathrm{prp}}}_{\widetilde{E}}(t', (\lceil \sigma/n \rceil + 2 \cdot (q_e + q_d))) + q_d \frac{2^{n-\tau}}{2^n - 1}$.

OCB MODE WITH VARIABLE-STRETCH SECURITY. We introduce ΘCBv (variable-stretch-ΘCB), a nonce-based AE scheme with variable stretch, obtained by slightly modifying ΘCB.

The tweakable blockcipher mode of operation ΘCBv is parameterized only by a tweakable blockcipher $\widetilde{E} : \mathcal{K} \times \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$. The tweak $\mathcal{T}$ is different than the one needed for ΘCB; it is of the form $\mathcal{T} = \mathcal{N} \times \mathcal{I}_T \times \mathbb{N}_0 \times \{0,1,2,3\} \cup \mathcal{I}_T \times \mathbb{N}_0 \times \{0,1,2,3\}$ where $\mathcal{I}_T \subseteq \{0,1,\ldots,n\}$ is the desired stretch-space of ΘCBv. The encryption and decryption algorithms of ΘCBv are exactly the same as those of ΘCB, that they now allow incorporate variable stretch and and that every call to $\widetilde{E}$ is now tweaked by τ, in addition to the other tweak components. Both algorithms are described in Figure 15. An illustration of the encryption algorithm is depicted in Figure 16.

Thanks to Theorem 2, establishing the $\mathbf{nvae}(\tau_c)$ security of ΘCBv requires little effort. The corresponding result is stated in Theorem 3.

Theorem 3. *Let $\widetilde{E} : \mathcal{K} \times \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$ be a tweakable blockcipher with $\mathcal{T} = \mathcal{N} \times \mathcal{I}_T \times \mathbb{N}_0 \times \{0,1,2,3\} \cup \mathcal{I}_T \times \mathbb{N}_0 \times \{0,1,2,3\}$. Then we have that*

$$\mathbf{Adv}^{\boldsymbol{nvae}(\tau_c)}_{\Theta\mathrm{CBv}[\widetilde{E}]}(t, \mathbf{q_e}, \mathbf{q_d}, \boldsymbol{\sigma}) \le \mathbf{Adv}^{\pm\widetilde{\mathrm{prp}}}_{\widetilde{E}}(t', q) + \sum_{\tau \in \mathcal{I}_T} \mathbf{Adv}^{\pm\widetilde{\mathrm{prp}}}_{\widetilde{E}}(t', q^\tau) + \mathbf{Adv}^{\pm\widetilde{\mathrm{prp}}}_{\widetilde{E}}(t', q^{\tau_c}) + q_d^{\tau_c} \cdot \frac{2^{n-\tau_c}}{2^n - 1}.$$

where $q^\tau = \lceil \sigma^\tau/n \rceil + 2 \cdot (q_e^\tau + q_d^\tau)$ for $\tau \in \mathcal{I}_T$, and $q = \sum_{\tau \in \mathcal{I}_T} q^\tau$, and $t' = t + O(\sigma)$ with $\sigma = \sum_{\tau \in \mathcal{I}_T} \sigma^\tau$.

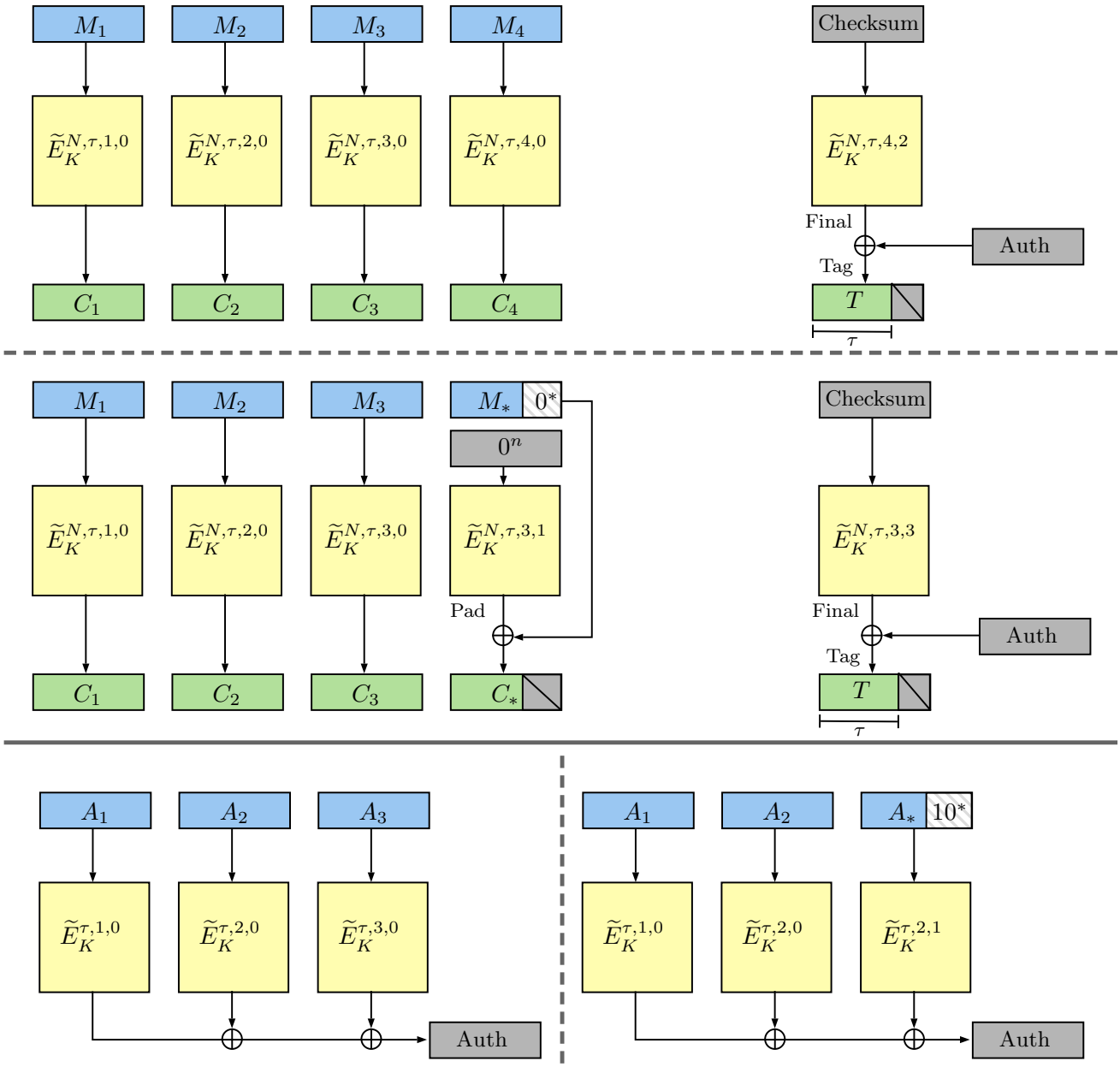

Fig. 16: Illustration of the encryption process of ΘCBv (inspired by [13]) instantiated with a tweakable blokcipher $\widetilde{E} : \mathcal{K} \times \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$. The top half depicts the encryption of a message with four complete blocks (top) with Sum= $\bigoplus_{i=1}^{4} M_i$ and the encryption of a message with three complete blocks and an incomplete block (bottom) with Sum= $\bigoplus_{i=1}^{3} \oplus M_* \| 10^*$. The bottom half of the picture shows processing of associated data of three complete blocks (left) or two complete blocks and an incomplete block (right).

Proof. We observe that if we fix the expansion value to τ_c in all queries, the nonce-based AE scheme $(\Theta\text{CBv}[\widetilde{E}])[\tau_c]$ that we get will be identical with the scheme $\Theta\text{CB}[\widetilde{E}, \tau_c]$. The result follows from this observation and the results of Lemmas 1 and 2 and Theorem 2. □

Lemma 2. *Let $\widetilde{E} : \mathcal{K} \times \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$ be a tweakable blockcipher with $\mathcal{T} = \mathcal{N} \times \mathcal{I}_T \times \mathbb{N}_0 \times \{0,1,2,3\} \cup \mathcal{I}_T \times \mathbb{N}_0 \times \{0,1,2,3\}$. Then we have that*

$$\mathbf{Adv}^{\boldsymbol{kess}}_{\Theta\text{CBv}[\widetilde{E}]}(t, \mathbf{q_e}, \mathbf{q_d}, \boldsymbol{\sigma}) \le \mathbf{Adv}^{\pm\widetilde{\text{prp}}}_{\widetilde{E}}(t', q) + \sum_{\tau \in \mathcal{I}_T} \mathbf{Adv}^{\pm\widetilde{\text{prp}}}_{\widetilde{E}}(t', q^\tau)$$

where $q^\tau = \lceil \sigma^\tau / n \rceil + 2 \cdot (q_e^\tau + q_d^\tau)$ for $\tau \in \mathcal{I}_T$, and $q = \sum_{\tau \in \mathcal{I}_T} q^\tau$, and $t' = t + O(\sigma)$ with $\sigma = \sum_{\tau \in \mathcal{I}_T} \sigma^\tau$.

Proof. Let $\mathscr{A}$ be a **kess** adversary with indicated resources. We proceed by replacing the tweakable blockcipher $\widetilde{E}$ by an ideal one, i.e. we sample an independent random tweakable permutation $\widetilde{\pi}_K \leftarrow\!\!\$\ \text{Perm}^{\mathcal{T}}(n)$ for every $K \in \mathcal{K}$ in both the **kess-R** and the **kess-I** game. The increase of $\mathscr{A}$'s advantage due to this replacement in the game **kess-R** is bounded by $\mathbf{Adv}^{\pm\widetilde{\text{prp}}}_{\widetilde{E}}(t, q)$ by a standard reduction. To bound the increase of $\mathscr{A}$'s advantage due to the replacement in the game **kess-I**, we observe that the replacement can be done gradually, for one value of stretch at a time. Thus, by a standard hybrid argument, the cumulative increase of advantage will be bounded by $\sum_{\tau \in \mathcal{I}_T} \mathbf{Adv}^{\pm\widetilde{\text{prp}}}_{\widetilde{E}}(t, q^\tau)$. Once $\widetilde{E}$ is replaced by a collection of random tweakable permutations in both games, we observe that in both games, the games will produce identical distributions. This is because both in **kess-R** and in **kess-I**, any two queries with any two unequal amounts of stretch τ_1 and τ_2 will be processed by two independent collections of random permutations (thanks to the separation of queries with different amounts of stretch by tweaks). □

Instantiating $\widetilde{E}$. In order to obtain a real-world scheme, we need to instantiate the tweakable blockcipher $\widetilde{E}$. The scheme OCB uses the XEX construction [23] that turns an ordinary blockcipher $E : \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ into a tweakable blockcipher $\widetilde{E} = \text{XEX}[E]$ with $\widetilde{E} : \mathcal{K} \times \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$. A call to $\widetilde{E} = \text{XEX}[E]$ is evaluated in two ways, depending on the tweak:

$$\widetilde{E}_K^{N,i,j}(X) = E_K(X \oplus \Delta_{N,i,j}) \oplus \Delta_{N,i,j}, \quad \text{or} \quad \widetilde{E}_K^{i,j}(X) = E_K(X \oplus \Delta_{i,j}).$$

In each call, the input (and in some cases also the output) of the blockcipher E is masked with special Δ-values, derived from the tweak and the secret key. An almost XOR universal hash $H : \mathcal{K} \times \{0,1\}^{<n} \to \{0,1\}^n$ with $H(K, N) = E_K(N \| 10^*)$ is used in the computation of the masking values.[5] In what follows, we silently represent binary strings and integers by element of $\text{GF}(2^n)$ whenever needed and do the multiplications in this field with some fixed representation. E.g. $2^2 \cdot (0^{n-2} \| 10)$ would return an n-bit string that represents the result of $x^2 \cdot x$ in $GF(2^n)$. The masking Δ-values are computed as follows:

$$\begin{aligned}
&\Delta_{N,0,0} = H(K, N),\\
&\Delta_{N,i+1,0} = \Delta_{N,i,0} \oplus L(\text{ntz}(i+1)) \text{ for } i \ge 0,\\
&\Delta_{N,i,j} = \Delta_{N,i,0} \oplus j \cdot L_* \text{ for } j \in \{0,1,2,3\},\\
&\Delta_{0,0} = 0^n,\\
&\Delta_{i+1,0} = \Delta_{i,0} \oplus L(\text{ntz}(i+1)) \text{ for } i \ge 0,\\
&\Delta_{i,j} = \Delta_{i,0} \oplus j \cdot L_* \text{ for } j \in \{0,1,2,3\},
\end{aligned}$$

where $L_* = E_K(0^n)$, $L(0) = 2^2 \cdot L_*$, $L(\ell) = 2 \cdot L(\ell - 1)$ for $\ell > 0$ and $\text{ntz}(i)$ denotes the number of trailing zeros in the binary representation of the integer i, e.g. $\text{ntz}(2) = 1$.

Lemma 3. *([23]) Let $E : \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ be a blockcipher and $\mathcal{T} = \mathcal{N} \times \mathbb{N}_0 \times \{0,1,2,3\} \cup \mathbb{N}_0 \times \{0,1,2,3\}$. Let $\mathscr{A}$ be an adversary that runs in time at most t, asks at most q queries, never asks queries with i-component exceeding 2^{n-5} and never asks decryption queries with tweaks from $\mathbb{N}_0 \times \{0,1,2,3\}$. Then*

$$\mathbf{Adv}^{\pm\widetilde{\text{prp}}^{\tau}}_{\text{XEX}[E]}(\mathscr{A}) \le \mathbf{Adv}^{\pm\text{prp}}_{E}(\mathscr{B}) + \frac{9.5q^2}{2^n}$$

for an adversary $\mathscr{B}$ that makes at most $2q$ queries and runs in time bounded by $t + O(q)$.

[5] A different AXU is used in the latest version of OCB [13], we opted for $E_K(\cdot)$ for the sake of simplicity.

EXTENDING THE TWEAKS WITH τ. In order to instantiate ΘCBv, we need to extend the tweaks of $\widetilde{E}$ with a fourth component: τ. To this end, we propose XEX′, which is obtained by a slight modification of the XEX construction. Informally, we expand the domain of the "j-part" of tweaks and represent it as $\mathcal{I}_T \times \{0,1,2,3\}$, compensating for this by decreasing the maximal value of i.

The tweakable blockcipher $\widetilde{E}' = \mathrm{XEX}'[E]$ is defined as follows. We again use the AXU $H(K,N)$. We uniquely label each element of $\mathcal{I}_T$ by an integer with a bijection $\lambda : \mathcal{I}_T \to \{0,1,\ldots,|\mathcal{I}_T|-1\}$. We define $m = \lceil \log_2 |\mathcal{I}_T| \rceil$, $L_* = E_K(0^n)$, $L_\tau = \lambda(\tau) \cdot 2^2 \cdot L_*$ for $\tau \in \mathcal{I}_T$, $L(0) = 2^{2+m} \cdot L_*$, and $L(\ell) = 2 \cdot L(\ell-1)$ for $\ell > 0$. The masking Δ-values are computed as follows:

$\Delta_{N,0,0,0} = H(K,N)$,
$\Delta_{N,\tau,0,0} = \Delta_{N,0,0,0} \oplus L_\tau$,
$\Delta_{N,\tau,i+1,0} = \Delta_{N,\tau,i,0} \oplus L(\mathrm{ntz}(i+1))$ for $i \geq 0$,
$\Delta_{N,\tau,i,j} = \Delta_{N,\tau,i,0} \oplus j \cdot L_*$ for $j \in \{0,1,2,3\}$,
$\Delta_{\tau,0,0} = L_\tau$,
$\Delta_{\tau,i+1,0} = \Delta_{\tau,i,0} \oplus L(\mathrm{ntz}(i+1))$ for $i \geq 0$,
$\Delta_{\tau,i,j} = \Delta_{\tau,i,0} \oplus j \cdot L_*$ for $j \in \{0,1,2,3\}$.

A call to $\widetilde{E}'$ is evaluated as follows:

$$\widetilde{E}'^{N,\tau,i,j}_K(X) = E_K(X \oplus \Delta_{N,\tau,i,j}) \oplus \Delta_{N,\tau,i,j}, \quad \text{or} \quad \widetilde{E}'^{\tau,i,j}_K(X) = E_K(X \oplus \Delta_{\tau,i,j}).$$

The security result for XEX′ construction is stated in Lemma 4.

Lemma 4. *Let $E : \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ be a blockcipher and $\mathcal{T} = \mathcal{N} \times \mathcal{I}_T \times \mathbb{N}_0 \times \{0,1,2,3\} \cup \mathcal{I}_T \times \mathbb{N}_0 \times \{0,1,2,3\}$ for some finite, non-empty $\mathcal{I}_T \subseteq \mathbb{N}_0$. Let $\mathscr{A}$ be an adversary that runs in time at most t, asks at most q queries, never asks queries with i-component exceeding $2^{n-(5+\lceil \log_2 |\mathcal{I}_T| \rceil)}$ and never asks decryption queries with tweaks from $\mathcal{I}_T \times N_0 \times \{0,1,2,3\}$. Then*

$$\mathbf{Adv}^{\pm\mathrm{prp}}_{\widetilde{\mathrm{XEX}'[E]}^{\mathcal{T}}}(\mathscr{A}) \leq \mathbf{Adv}^{\pm\mathrm{prp}}_E(\mathscr{B}) + \frac{9.5q^2}{2^n}$$

for an adversary $\mathscr{B}$ that makes at most $2q$ queries and runs in time bounded by $t + O(q)$.

The treatment of τ-tweak component in XEX′ construction is equivalent to a one where we would injectively encode τ, j into a single integer $j' = 2^2\tau + j$. Similar approach has been used in [19, 20], where it is shown that the essential properties of the mollows:asking values necessary for the security proof of [23] are preserved. The same arguments apply here, so we omit the proof of Lemma 4.

OCBv: PRACTICAL AE WITH VARIABLE STRETCH. We define the blockcipher mode OCBv, a nonce based AE scheme with variable stretch. OCBv is only parameterized by a blockcipher E. It is obtained by instantiating the tweakable blockcipher in ΘCBv by the XEX′ costruction, i.e. $\mathrm{OCBv}[E] = \Theta\mathrm{CBv}[\mathrm{XEX}'[E]]$ and its security is analysed in Theorem 4.

Theorem 4. *Let $\widetilde{E} : \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ be a blockcipher. We have that*

$$\mathbf{Adv}^{\boldsymbol{nvae}(\tau_c)}_{\mathrm{OCBv}[E]}(t, \mathbf{q_e}, \mathbf{q_d}, \boldsymbol{\sigma}) \leq \mathbf{Adv}^{\pm\mathrm{prp}}_E(t', 2q) + \sum_{\tau \in \mathcal{I}_T} \mathbf{Adv}^{\pm\mathrm{prp}}_E(t', 2q^\tau) + \mathbf{Adv}^{\pm\mathrm{prp}}_E(t', 2q^{\tau_c}) + \frac{28.5q^2}{2^n} + q_d^{\tau_c} \frac{2^{n-\tau_c}}{2^n - 1}.$$

where $q^\tau = \lceil \sigma^\tau / n \rceil + 2 \cdot (q_e^\tau + q_d^\tau)$ for $\tau \in \mathcal{I}_T$, and $q = \sum_{\tau \in \mathcal{I}_T} q^\tau$ and $t' = t + O(\sigma)$ with $\sigma = \sum_{\tau \in \mathcal{I}_T} \sigma^\tau$.

If we further assume that the $\mathbf{Adv}^{\pm\mathrm{prp}}_E$ is non-decreasing w.r.t. both q and t, then we can further simplify the bound to the form

$$\mathbf{Adv}^{\mathbf{nvae}(\tau_c)}_{\mathrm{OCBv}[E]}(t, \mathbf{q_e}, \mathbf{q_d}, \boldsymbol{\sigma}) \leq (|\mathcal{I}_T| + 2) \cdot \mathbf{Adv}^{\pm\mathrm{prp}}_E(t', 2q) + \frac{28.5q^2}{2^n} + q_d^{\tau_c} \cdot \frac{2^{n-\tau_c}}{2^n - 1}.$$

Proof. The result in Theorem 4 follows from Theorem 3 and Lemma 4 by applying triangle inequality on the terms that arise from applying Lemma 4. □

PERFORMANCE OF OCBV. The performance of OCBv can be expected to be very similar to that of OCB, as the two schemes only differ in the way the masking Δ-values are computed. In addition to the operations necessary to compute Δ-offsets in OCB, the computation of the L_τ-values has to be done for OCBv. However, these can be precomputed at the initialization phase and stored, so the cost of their computation will be amortized over all queries. The only additional processing that remains after dealing with L_τ-s is a single xor of a precomputed L_τ to a Δ-value, necessary in every query. This is unlikely to impact the performance significantly.

References

1. 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. LNCS, vol. 8873, pp. 105–125. Springer (2014)
2. Bellare, M., Namprempre, C.: Authenticated Encryption: Relations among Notions and Analysis of the Generic Composition Paradigm. In: Okamoto, T. (ed.) ASIACRYPT 2000. LNCS, vol. 1976, pp. 531–545. Springer (2000)
3. 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 (2000)
4. Bernstein, D.J.: Cryptographic competitions: CAESAR. http://competitions.cr.yp.to
5. Bernstein, D.J.: Cryptographic competitions: Disasters. https://competitions.cr.yp.to/disasters.html
6. Eichlseder, M.: Remark on variable tag lengths and OMD. `crypto-competitions` mailing list. April 25, 2014.
7. Fleischmann, E., Forler, C., Lucks, S.: McOE: A Family of Almost Foolproof On-Line Authenticated Encryption Schemes. In: Canteaut, A. (ed.) FSE 2012. LNCS, vol. 7549, pp. 196–215. Springer (2012)
8. Fuhr, T., Leurent, G., Suder, V.: Collision Attacks Against CAESAR Candidates - Forgery and Key-Recovery Against AEZ and Marble. In: Iwata, T., Cheon, J.H. (eds.) ASIACRYPT 2015. LNCS, vol. 9453, pp. 510–532. Springer (2015)
9. 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. LNCS, vol. 9056, pp. 15–44. Springer (2015)
10. Hoang, V.T., Reyhanitabar, R., Rogaway, P., Vizár, D.: Online Authenticated-Encryption and its Nonce-Reuse Misuse-Resistance. In: Gennaro, R., Robshaw, M. (eds.) CRYPTO 2015. LNCS, vol. 9215, pp. 493–517. Springer (2015)
11. Iwata, T.: CLOC and SILC will be tweaked. `crypto-competitions` mailing list. August 4, 2015.
12. Katz, J., Yung, M.: Unforgeable Encryption and Chosen Ciphertext Secure Modes of Operation. In: Schneier, B. (ed.) FSE 2000. LNCS, vol. 1978, pp. 284–299. Springer (2001)
13. Krovetz, T., Rogaway, P.: The Software Performance of Authenticated-Encryption Modes. In: Joux, A. (ed.) FSE 2011. LNCS, vol. 6733, pp. 306–327. Springer (2011)
14. Manger, J.H.: [Cfrg] Attacker changing tag length in OCB. IRTF `CFRG` mailing list. May 29, 2013.
15. Minematsu, K.: AES-OTR v2. `crypto-competitions` mailing list. August 31, 2015.
16. Namprempre, C., Rogaway, P., Shrimpton, T.: Reconsidering Generic Composition. In: Nguyen, P.Q., Oswald, E. (eds.) EUROCRYPT 2014. Lecture Notes in Computer Science, vol. 8441, pp. 257–274. Springer (2014)
17. Nandi, M.: RE:CLOC and SILC will be tweaked. `crypto-competitions` mailing list. August 5, 2015.
18. Reyhanitabar, R.: OMD version 2: a tweak for the 2nd round. `crypto-competitions` mailing list. August 27, 2015.
19. Reyhanitabar, R., Vaudenay, S., Vizár, D.: Misuse-Resistant Variants of the OMD Authenticated Encryption Mode. In: Chow, S.S.M., Liu, J.K., Hui, L.C.K., Yiu, S. (eds.) ProvSec 2014. Lecture Notes in Computer Science, vol. 8782, pp. 55–70. Springer (2014)
20. Reyhanitabar, R., Vaudenay, S., Vizár, D.: Boosting OMD for Almost Free Authentication of Associated Data. In: Leander, G. (ed.) FSE 2015. LNCS, vol. 9054, pp. 411–427. Springer (2015)
21. Rogaway, P.: Re: [Cfrg] Attacker changing tag length in OCB. IRTF `CFRG` mailing list. Jun 3, 2013.
22. Rogaway, P.: Authenticated-Encryption with Associated-Data. In: ACM CCS 2002. pp. 98–107 (2002)
23. Rogaway, P.: Efficient Instantiations of Tweakable Blockciphers and Refinements to Modes OCB and PMAC. In: Lee, P.J. (ed.) ASIACRYPT 2004. LNCS, vol. 3329, pp. 16–31. Springer (2004)
24. Rogaway, P.: Nonce-Based Symmetric Encryption. In: Roy, B.K., Meier, W. (eds.) FSE 2004. LNCS, vol. 3017, pp. 348–359. Springer (2004)
25. Rogaway, P., Bellare, M., Black, J., Krovetz, T.: OCB: A Block-Cipher Mode of Operation for Efficient Authenticated Encryption. In: ACM CCS 2001. pp. 196–205 (2001)
26. Rogaway, P., Shrimpton, T.: A Provable-Security Treatment of the Key-Wrap Problem. In: EUROCRYPT 2006. pp. 373–390 (2006)
27. Rogaway, P., Shrimpton, T.: Deterministic authenticated-encryption: A provable-security treatment of the key-wrap problem. IACR Cryptology ePrint Archive 2006, 221 (2006)
28. Rogaway, P., Wagner, D.: A Critique of CCM. IACR Cryptology ePrint Archive 2003, 70 (2003)
29. Struik, R.: AEAD ciphers for highly constrained networks. DIAC 2013 presentation, August 13, 2013

A A short guide to nvAE

Interpretation of the nvAE security advantage. The notion of $\mathbf{nvae}(\tau_c)$ is parameterized by a constant, but arbitrary amount of stretch τ_c from the stretch space $\mathcal{I}_T$ of the AE scheme Π in question. In the $\mathbf{nvae}(\tau_c)$-$\mathbf{I}_\Pi$ security game, only queries expanded by τ_c bits will be subjected to "idealization". For all other expansions, we give the adversary complete freedom to ask any queries it wants (except for the nonce-requirement), but their behaviour is the same in both security games. An $\mathbf{nvae}(\tau_c)$ security bound that assumes no particular value or constraint for τ_c will therefore tell us, what security guarantees can we expect from queries stretched by τ_c bits specifically, for any $\tau_c \in \mathcal{I}_T$.

Looking at the security bound itself, we are able to tell if there are any undesirable interactions between queries with different amounts of stretch. This is best illustrated by revisiting the problems and forgery attack from Sections 1 and 3 in the **nvae**(τ_c) security model.

Attacks in nvAE model. Consider the original, unmodified scheme OCB [13], that produces the tag by truncating an n-bit (with $n > \tau$) to τ bits. In case of simultaneous use of two (or more) amounts of stretch $\tau_1 < \tau_2$ with the same key, we can forge a ciphertext stretched by τ_1 bits by τ_2-bit-stretched ciphertext truncation. This would correspond to an attack with an **nvae**(τ_1) advantage of $1 - 2^{-\tau_1}$ and constant resources.

If the same scheme is treated with the heuristic measures, i.e.nonce-stealing, and encoding τ in AD, from Section 3 (let's call it hOCB), we consider the forgery attack from the same Section. Assume that there are four instances of hOCB, with $32, 64, 96$ and 128 bit tags. To make a forgery with 128-bit tag, we have to find a forgery with 32 bits and then exhaustively search for three 32-bit extensions of this forgery. This gives us an **nvae**(128) advantage equal to 1, requiring 4 encryption queries, $3 \cdot 2^{32}$ verification queries with stretch other than 128 bits and 2^{32} verification stretched by 128 bits. The effort necessary for such a forgery is clearly smaller than we could hope for.

"Good" bounds. After seeing examples of attacks, one may wonder: what kind of **nvae**(τ_c) security bound should we expect from a secure nvAE scheme? For every scheme, it must be always possible to guess a ciphertext with probability $2^{-\tau_c}$. Thus the bound must always contain a term of the form $c \cdot (q_d^{\tau_c})^\alpha / 2^{\tau_c}$ for some positive constants c and α, or something similar.

Even though the security level for τ_c-stretched queries should be independent of any other queries, it is usually unavoidable to have a gradual increase of advantage with every query made by the adversary. This increase can generally depend on all of the adversarial resources, but should not depend on τ_c itself.

An example of a secure scheme's **nvae**(τ_c) bound can be found in Theorem 4. It consist of the fraction $(q_d^{\tau_c} \cdot 2^{n-\tau_c})/(2^n - 1) \approx q_d^{\tau_c}/2^{\tau_c}$, advantage bounds for the used blockcipher and a birthday-type term that grows with the total amount of data processed. We see, that queries stretched by $\tau \neq \tau_c$ bits will not unexpectedly increase adversary's chances to break OCBv, and that the best attack strategy is indeed issuing decryption queries with τ_c bits of stretch.