


CCA Security with Short AEAD Tags

Mustafa Khairallah 

Seagate Research Group, Singapore, Singapore

Department of Electrical and Information Technology, Lund University, Lund, Sweden

Abstract. The size of the authentication tag represents a significant overhead for applications that are limited by bandwidth or memory. Hence, some authenticated encryption designs have a smaller tag than the required privacy level, which was also suggested by the NIST lightweight cryptography standardization project. In the ToSC 2022, two papers have raised questions about the IND-CCA security of AEAD schemes in this situation. These papers show that (a) online AE cannot provide IND-CCA security beyond the tag length, and (b) it is possible to have IND-CCA security beyond the tag length in a restricted Encode-then-Encipher framework.

In this paper, we address some of the remaining gaps in this area. Our main result is to show that, for a fixed stretch, Pseudo-Random Injection security implies IND-CCA security as long as the minimum ciphertext size is at least as large as the required IND-CCA security level. We also show that this bound is tight and that any AEAD scheme that allows empty plaintexts with a fixed stretch cannot achieve IND-CCA security beyond the tag length.

Next, we look at the weaker notion of MRAE security, and show that two-pass schemes that achieve MRAE security do not achieve IND-CCA security beyond the tag size. This includes SIV and rugged PRPs.

Keywords: Chosen Ciphertext Attacks · IND-CCA · AEAD · SIV · Authentication · Rugged PRP

1 Introduction

Authenticated Encryption with Associated Data (AEAD) is one of the most important symmetric-key primitives. It provides privacy and authenticity, simultaneously. It has gained significant attention over the past 25 years, culminating in two cryptographic competitions to either recommend or standardize AEAD schemes for a variety of applications; the CAESAR competition [com19] and the NIST lightweight cryptography standardization project [oST19]. While these projects encouraged cryptographers and designers to diversify the design space of AEADs, they also helped shine light on some of the less studied aspects of AEADs. AEAD schemes typically require a nonce N or a random IV , usually communicated out-of-band, and expand the ciphertext size by λ bits, referred to as the ciphertext stretch, or *stretch* for short. The integrity of an AEAD scheme cannot be ensured with $\lambda = 0$, and in terms of bit-security, it is capped at λ bits, as the adversary can simply try to guess the redundancy in the ciphertext. However, the impact of the stretch on privacy has been an interesting aspect of the security of AEAD schemes.

Security Notions The security notions of AEAD schemes can be defined using indistinguishability games between a real world and an ideal world. An AEAD scheme consists of

E-mail: mustafa.khairallah.1608@eit.lth.se (Mustafa Khairallah)

This work is licensed under a “CC BY 4.0” license.

Date of this document: 2024-03-27.



an encryption algorithm and a decryption/verification algorithm. The security notions are also defined using two idealized oracles; $\$$ and \perp . The first oracle replaces the encryption algorithm and returns random strings, while the latter replaces the decryption algorithm and rejects all ciphertexts. The game is defined between a challenger and an adversary, where the challenger flips a coin at the beginning of the game, and decides to operate in the real world or the ideal world. The adversary makes q_e queries to the left oracle and q_d queries to the right oracle. It runs in time t . It returns 1 if it thinks the challenger is ideal. For a security notion \mathbf{n} , a scheme Π and an adversary \mathbf{A} , the advantage of \mathbf{A} against the challenger is defined as

$$\mathbf{Adv}_{\Pi}^{\mathbf{n}}(\mathbf{A}) \stackrel{\text{def}}{=} |\Pr[\mathbf{A}^{\text{Real}} \Rightarrow 1] - \Pr[\mathbf{A}^{\text{Ideal}} \Rightarrow 1]|.$$

Table 1 sums up the real and ideal oracles corresponding to four prominent security notions. IND-CPA refers to indistinguishability against chosen plaintext adversaries, while IND-CCA refers to indistinguishability against chosen ciphertext adversaries. INT-CTXT refers to integrity of ciphertexts (regardless of confidentiality) and AEAD refers to a unified security notion of both IND-CPA and INT-CTXT¹. We are mainly interested in IND-CCA security, which captures confidentiality even when the decryption oracle is always real and can be occasionally forged.

Table 1: The real-world and ideal-world oracles corresponding to different AEAD security notions

Notion	Real World	Ideal World
ind-cpa	Enc, \perp	$\$, \perp$
int-ctxt	Enc, Dec	Enc, \perp
aead	Enc, Dec	$\$, \perp$
ind-cca	Enc, Dec	$\$, \text{Dec}$

Related Work The security notions of AEADs have been heavily studied. We give a few highlights that are relevant to our study. In 2000, Bellare and Namprempre [BN00] studied the relation between different security notions of AEAD, and one of their results is to show that IND-CPA and INT-CTXT together imply IND-CCA. In particular, if there is an IND-CCA adversary \mathbf{A} against an AEAD scheme Π that makes at most q_e encryption queries and q_d forgery attempts, and runs in time at most t , then there exist two adversaries \mathbf{B} and \mathbf{C} , such that

$$\mathbf{Adv}_{\Pi}^{\text{ind-cca}}(\mathbf{A}) \leq \mathbf{Adv}_{\Pi}^{\text{ind-cpa}}(\mathbf{B}) + 2 \cdot \mathbf{Adv}_{\Pi}^{\text{int-ctxt}}(\mathbf{C}),$$

where both adversaries run in time $O(t)$, \mathbf{B} makes at most q_e encryption queries, and \mathbf{C} makes at most q_e encryption queries and q_d forgery attempts. While their result is for a weaker security notion, namely indistinguishability of the encryption of random plaintexts, the result is still significant and motivates our study, among others.

During the CAESAR competition, Hoang *et al.* [HKR15] proposed AEZ, an AEAD scheme aimed at being a secure AEAD scheme with arbitrary stretch sizes. In fact, they consider λ as an input from the user/adversary. In order to achieve this, they designed an enciphering scheme. An enciphering scheme is a variable-block-size Tweakable Block Cipher (TBC). Then, they apply a framework known as Encode-then-Encipher (EtE), where the message is encoded with λ bits of redundancy, *e.g.*, the message can be padded with 0^λ , and then encrypted using the enciphering scheme. This approach has a lot of promise, but their scheme uses an internal fixed-block size TBC with block size of n bits.

¹While there are other security notions, such as release of unverified plaintexts, they are not relevant to this work.

The scheme has birthday bound security, capping the security to $n/2$ bits. Besides, it follows the *proof-then-prune* strategy, where the security proof is done assuming an ideal TBC, but in practice they use a 4-round cipher based on the AES round function. This approach has been shown to lead to birthday-bound key recovery attacks [CG16] and plays a significant role in the scheme’s excellent practical performance. Nevertheless, we view their work on defining the robust AE security notion as essential to our work. We will be referring to a more restricted version known as Pseudo-Random Injection (PRI), where we consider IND-CCA security when λ can be short, but is fixed as part of the scheme. In 2016, Abed *et al.* [AFL⁺16] proposed the Robust IV (RIV) scheme as more efficient solution to address similar AEAD goals to AEZ, but the authors provide a security bound where the AEAD security is upper bounded by the INT-CTXT security.

At CRYPTO 2022, Degabriele and Karadžić [DK22] proposed the concept of a rugged Pseudo-Random Permutation (PRP); a variable-block-length PRP scheme that is not secure against IND-CCA attacks, but can be used in the EtE framework when part of the encoded plaintext text is encrypted using an IND-CCA-secure PRP. Then, they propose two EtE-like AEAD schemes based on their construction, one with λ -bit AEAD security and one with $\lambda/2$ -bit AEAD security.

Last but not least, two recent papers appeared in ToSC22 addressing the IND-CCA security of online AEAD schemes. An online AEAD scheme is a scheme that is parameterized by a small integer m , where the first m bits of the ciphertext depend only on the first m bits of plaintext; the first $2m$ bits of ciphertext depend on the first $2m$ bits of plaintext;...*etc.* In [Kha22], Khairallah showed that online AEAD schemes cannot have IND-CCA security more than λ bits. He also showed that the Combined Feedback (COFB) [CIMN20] AEAD scheme has at most $n/2$ -bit IND-CCA security when instantiated with an n -bit block cipher. In [HII⁺22], Hosoyamada *et al.* showed a similar result and provided an attack on the ROCCA AEAD scheme [SLN⁺21] with 2^λ queries. They also showed that it is possible to build a nonce-based AEAD scheme using the EtE framework with 256-bit IND-CCA security and 128-bit INT-CTXT security, if the underlying enciphering scheme is a fixed length strong Tweakable PRP (TPRP).

Contributions In this paper, we start from the same question in [HII⁺22]: *Can we design an AEAD scheme with 256-bit IND-CCA security and 128-bit INT-CTXT security?* But we expand on the problem, and redefine it as follows:

Can we have an AEAD scheme with λ -bit INT-CTXT security and more than λ -bit IND-CCA security?

Table 2: Different types of AEAD schemes and whether they can achieve IND-CCA security beyond the tag length. "Yes" requires a sufficiently large minimum plaintext length

Definition	CCA Security $> \lambda$ bits	Ref
Online AE	No	[Kha22]
Encode-then-Encipher	Yes	[HII ⁺ 22]
MRAE with online encryption	No	This work
Rugged PRP	No	This work
PRI	Yes	This work

To address this question, we contribute multiple results:

1. While typical practical AEAD schemes can encrypt arbitrary messages, including empty string, we show a negative result that no AEAD scheme can have IND-CCA security more than λ bits when empty strings are part of the plaintext space and λ

is fixed. More generally, we show that any AEAD scheme can have at most $s + \lambda$ IND-CCA security when λ is fixed and s is the minimum plaintext size, in the nonce respecting model, and the security of that scheme degrades linearly with the number of nonce repetitions. We show this by modelling the best possible fixed-stretch AEAD scheme, dubbed as Pseudo-Random Injection (PRI), and proving an upper bound on the adversarial IND-CCA advantage and a matching attack.

2. Since [Kha22] and [HII⁺22] address online AEAD, a natural next step is to study two-pass AEAD. The most popular such scheme is the Synthetic IV (SIV) scheme proposed by Rogaway and Shrimption [RS06]. Peyrin and Seurin proposed a nonce-based generalization of SIV [PS16], dubbed nSIV. We show an IND-CCA attack on SIV when the underlying encryption scheme is IND-CCA-insecure, and show that when the encryption scheme is stream-cipher-like, the attack can be used to decrypt any messages, capping the IND-CCA security of SIV in these cases to λ bits.
3. We also show that while the recently proposed rugged PRPs [DK22] share a similar philosophy to this paper, they are not sufficient to address our security goals. We do this by proposing matching attacks in the IND-CCA model.

While our results are mostly negative, they help paint a clearer picture of the IND-CCA security with short tags landscape. We show that two-pass schemes do not achieve the required security in this model and that the enciphering assumption in [HII⁺22] is not needed. However, the PRI assumption we need is very close to the enciphering assumption. Besides, we show -by both security proof and attack- that in order to achieve the required security, we must constrain the minimum size of the plaintext. It is possible to have a different security argument based on variable stretch size and the minimum ciphertext size in general, but we leave that approach out of our scope. Table 2 shows a summary of the implications of [Kha22, HII⁺22] and our work.

2 Preliminaries

Notations We use small case letters, *e.g.*, v , to refer to integer variables. We use uppercase letters, *e.g.*, V , to refer to variables that are bit-strings. We use calligraphic letters, *e.g.*, \mathcal{V} , to refer to sets of values. We use boldface uppercase letters, *e.g.*, \mathbf{V} , to refer to adversaries. ε refers to an empty bit-string. $\{0, 1\}^b$ is the set of all bit-strings of size exactly b . $\{0, 1\}^{b+}$ is the set of all bit-strings of greater than or equal b . $\{0, 1\}^* \equiv \{0, 1\}^{0+}$ is the set of all bit-strings including ε . For two bit-strings X and Y , $X||Y$ is the concatenation of X and Y . $|X|$ is the length of the bit-string X expressed by the number of bits. $[X]_n$ is the bit-string composed of the n -leftmost bits of X , while $\lceil X \rceil_n$ is the bit-string composed of n -rightmost bits of X . \leftarrow is an assignment from a statement on the right hand side to a variable on the left hand side. $\xleftarrow{\$}$ samples a value uniformly from the set on the right hand side and assigns it to the variable on the left hand side. $\mathbf{X} \Rightarrow X$ means the algorithm/adversary on the left hand side returns X .

Authenticated Encryption An Authenticated Encryption (AE) scheme [BN00] is a symmetric-key algorithm that provides both privacy and authenticity. An AEAD scheme [Rog02] differs in that both algorithms take an extra input called associated data A which is a public portion of the message, used for authentication only. In this paper, we focus on nonce-based AEAD, sometimes known as NAE, which takes a third input, N , called nonce. An AEAD scheme Π is a triplet $(\mathcal{K}, \mathbf{Enc}, \mathbf{Dec})$, where \mathcal{K} is the key space, $\mathbf{Enc} : \mathcal{K} \times \mathcal{N} \times \mathcal{A} \times \mathcal{M} \rightarrow \mathcal{C}$ is the encryption algorithm and $\mathbf{Dec} : \mathcal{C} \times \mathcal{N} \times \mathcal{A} \times \mathcal{C} \rightarrow \mathcal{M} \cup \{\perp\}$ is the decryption algorithm, which returns \perp if the input is not a valid ciphertext. Let $\mathcal{M} = \{0, 1\}^{s+}$, then $\mathcal{C} = \{0, 1\}^{(s+\lambda)+}$, where s is the minimum-plaintext size and λ is the

ciphertext stretch. For every $\mathbf{Enc}(K, N, A, M) \Rightarrow C$, $|C| = |M| + \lambda$. For some schemes, the ciphertext can be separated into two distinctive bit-strings, in which case we redefine $C \equiv \mathcal{T} \times C'$, where $C = T \| C'$, $|T| = \lambda$ and $|C'| = |M|$. In this case, T is referred to as the tag, and $|T|$ is the stretch. In some scheme, we define a tweak space $\mathcal{T} = \mathcal{N} \times \mathcal{A}$ and we combine each pair (N, A) in one variable T_w . In such cases, we refer to T_w as the nonce and we count how many times a nonce repeats accordingly.

Let $\mathcal{S} : \mathcal{N} \times \mathcal{A} \times \mathcal{M} \rightarrow \mathcal{C}$ be a random oracle that returns a uniformly random ciphertext of the same size as \mathbf{Enc} and $\perp : \mathcal{N} \times \mathcal{A} \times \mathcal{C} \rightarrow \{\perp\}$ is an oracle that rejects all ciphertexts. Let \mathbf{A} be an adversary against Π . We define four security notions:

$$\mathbf{Adv}_{\Pi}^{\text{ind-cpa}}(\mathbf{A}) \stackrel{\text{def}}{=} |\Pr[K \xleftarrow{\mathcal{S}} \mathcal{K} : \mathbf{A}^{\mathbf{Enc}, \perp} \Rightarrow 1] - \Pr[\mathbf{A}^{\mathcal{S}, \perp} \Rightarrow 1]|$$

$$\mathbf{Adv}_{\Pi}^{\text{int-ctxt}}(\mathbf{A}) \stackrel{\text{def}}{=} |\Pr[K \xleftarrow{\mathcal{S}} \mathcal{K} : \mathbf{A}^{\mathbf{Enc}, \text{Dec}} \Rightarrow 1] - \Pr[K \xleftarrow{\mathcal{S}} \mathcal{K} : \mathbf{A}^{\mathbf{Enc}, \perp} \Rightarrow 1]|$$

$$\mathbf{Adv}_{\Pi}^{\text{aead}}(\mathbf{A}) \stackrel{\text{def}}{=} |\Pr[K \xleftarrow{\mathcal{S}} \mathcal{K} : \mathbf{A}^{\mathbf{Enc}, \text{Dec}} \Rightarrow 1] - \Pr[\mathbf{A}^{\mathcal{S}, \perp} \Rightarrow 1]|$$

$$\mathbf{Adv}_{\Pi}^{\text{ind-cca}}(\mathbf{A}) \stackrel{\text{def}}{=} |\Pr[K \xleftarrow{\mathcal{S}} \mathcal{K} : \mathbf{A}^{\mathbf{Enc}, \text{Dec}} \Rightarrow 1] - \Pr[K \xleftarrow{\mathcal{S}} \mathcal{K} : \mathbf{A}^{\mathcal{S}, \text{Dec}} \Rightarrow 1]|$$

The last one captures IND-CCA security and is the focus of our work.

3 CCA Security of Pseudo-Random Injections

In this section, we study the IND-CCA security of a Pseudo-Random Injection (PRI). We start by defining tweakable PRIs. Then, we define their security. We refer to one of the results of [RS06] that addresses the security gap between PRIs when the tweak is fixed and ideal Deterministic AEAD (DAE). While the ideal AEAD scheme is defined according to AEAD security, we can argue that tweakable PRIs capture the best possible notion of AEAD in practice. An ideal AEAD scheme rejects all decryption queries that have not been generated by its encryption random oracle. In practice, an AEAD scheme should be able to decrypt valid ciphertexts that have not been generated yet.

Definition 1. A tweakable fixed-stretch injection is a keyed function $\tilde{\pi} : \mathcal{K} \times \mathcal{T} \times \{0, 1\}^{s+} \rightarrow \{0, 1\}^{(s+\lambda)+}$, where \mathcal{K} is the key space, \mathcal{T} is the tweak space, s is the minimum plaintext length expressed in bits, and λ is referred to as the ciphertext stretch, such that $\tilde{\pi}(K, T_w, \cdot)$ is an injective function from $\{0, 1\}^{s+}$ to $\{0, 1\}^{(s+\lambda)+}$. $\forall M \in \{0, 1\}^{s+}, K \in \mathcal{K}, T_w \in \mathcal{T}$, $|\tilde{\pi}(K, T_w, M)| = |M| + \lambda$. We sometimes refer to $\tilde{\pi}(K, T_w, M)$ as $\tilde{\pi}_K^{T_w}(M)$.

Definition 2. The security of a tweakable fixed-stretch injection $\tilde{\pi} : \mathcal{K} \times \mathcal{T} \times \{0, 1\}^{s+} \rightarrow \{0, 1\}^{(s+\lambda)+}$ for a given key K selected randomly from \mathcal{K} is defined by its indistinguishability from a tweakable fixed-stretch injection f selected randomly from the set of injections with the same domain, co-domain and stretch. $f : \mathcal{T} \times \{0, 1\}^{s+} \rightarrow \{0, 1\}^{(s+\lambda)+}$ is a set of independent uniform random fixed-stretch injections indexed by the tweak $T_w \in \mathcal{T}$. Let $\mathbf{Adv}_{\tilde{\pi}}^{\text{pri}}(\mathbf{A})$ denote the Pseudo-Random Injection (PRI) advantage of $\tilde{\pi}$ against an adversary \mathbf{A} . It is defined as

$$\mathbf{Adv}_{\tilde{\pi}}^{\text{pri}}(\mathbf{A}) \stackrel{\text{def}}{=} |\Pr[K \xleftarrow{\mathcal{S}} \mathcal{K} : \mathbf{A}^{\tilde{\pi}(K, \cdot, \cdot), \tilde{\pi}^{-1}(K, \cdot, \cdot)} \Rightarrow 1] - \Pr[\mathbf{A}^{f(\cdot, \cdot), f^{-1}(\cdot, \cdot)} \Rightarrow 1]|$$

where $\tilde{\pi}^{-1}(K, T_w, C)$ returns M if $\tilde{\pi}(K, T_w, M) = C$, and \perp if no such point exists, and similarly for f^{-1} .

We note that PRI security is similar to robust AE security. However, the two differ in that robust AE considers a variable stretch that is given by the user/adversary as an input, as pointed out in [HKR15]. In [RS06], the authors showed that when the tweak is fixed, a PRI is almost an ideal DAE. However, the bound is quite large when the stretch is short.

Theorem 1. ([RS06, Theorem 7]) Let $\tilde{\pi} : \mathcal{K} \times \mathcal{T} \times \{0, 1\}^{s+} \rightarrow \{0, 1\}^{(s+\lambda)+}$ be a PRI. Let \mathbf{A} be an adversary that makes at most q_e queries to $\tilde{\pi}$ and q_d queries to $\tilde{\pi}^{-1}$, with a total of $q = q_e + q_d$ queries. Then,

$$|\mathbf{Adv}_{\tilde{\pi}}^{\text{pri}}(\mathbf{A}) - \mathbf{Adv}_{\tilde{\pi}}^{\text{dae}}(\mathbf{A})| \leq \frac{q^2}{2^{s+\lambda+1}} + \frac{4q_d}{2^\lambda}.$$

In order to get a better and more practical bound, we consider the restricted case of nonce-misusing adversaries, where the adversary makes at most μ encryption queries with the same tweak T_w and consider IND-CCA security rather than AEAD security. The next theorem is a generalization of Theorem 1 in [HII⁺22]. It is also influenced by and uses techniques from Theorem 7 of [RS06]. Afterwards, we give a matching attack.

Theorem 2. Let $\tilde{\pi} : \mathcal{K} \times \mathcal{T} \times \{0, 1\}^{s+} \rightarrow \{0, 1\}^{(s+\lambda)+}$ be a PRI. Then, for any IND-CCA adversary \mathbf{A} that makes q_e queries to $\tilde{\pi}$ and q_d queries to $\tilde{\pi}^{-1}$, there exists a PRI adversary \mathbf{B} that makes at most q_e queries to $\tilde{\pi}$ and q_d queries to $\tilde{\pi}^{-1}$ and runs in time at most $O(t)$, such that

$$\mathbf{Adv}_{\tilde{\pi}}^{\text{ind-cca}}(\mathbf{A}) \leq \mathbf{Adv}_{\tilde{\pi}}^{\text{pri}}(\mathbf{B}) + \frac{(\mu - 1)q_e}{2^{s+\lambda}} + \frac{5\mu q_d}{2^{s+\lambda}} + \frac{2q_d}{2^{s+\lambda}},$$

given $(q_d + q_e) < 2^{s+\lambda-1}$. For any $T_w \in \mathcal{T}$, \mathbf{A} makes at most μ queries to $\tilde{\pi}$.

Proof. First, we replace $\tilde{\pi}$ with an ideal random injection, which gives us the first term. Next, we describe the PRI oracles and the ideal world oracles, then we describe three bad events where that are used to analyze the adversarial advantage. The oracles of both worlds are described in Algorithm 1.

Adversarial Queries: The adversary makes queries to its two oracle with the following conditions: it does not repeat queries and does not forward queries, *i.e.* if $C \leftarrow \mathbf{Enc}(T_w, M)$, the adversary cannot make the query $\mathbf{Dec}(T_w, C)$ and if $M \leftarrow \mathbf{Dec}(T_w, C)$, the adversary cannot make the query $\mathbf{Enc}(T_w, M)$.

PRI oracles: The oracles of the PRI are the oracles of Algorithm 1 including the highlighted lines. They are based on the PRI oracles proposed in [RS06, Theorem 7]. These oracles are a implementation of a random injection with the desired domain and range, using lazy sampling. The first oracle \mathbf{Enc} selects a valid image for the injection with tweak T_w and the input M . For each tweak T_w and ciphertext length c , it maintains a list $\mathcal{V}^{T_w, c}$ of queried inputs. The second oracle \mathbf{Dec} simulates a decryption function. It determines the list of valid ciphertexts and the list of valid plaintexts. Valid ciphertexts are ciphertexts that have not generated by an \mathbf{Enc} query and have not been queried to \mathbf{Dec} . The list of valid plaintexts are plaintexts that have not been queried to \mathbf{Enc} or have not been generated by \mathbf{Dec} and are not equal to \perp . The \mathbf{Dec} oracle then samples a biased coin with bias $|\mathcal{M}^e|/|\mathcal{C}^e|$ based on the sizes of these two lists and decides whether the forgery is valid or not. If the forgery is valid, it sames a valid plaintext and assigns it as a corresponding input. If not, it outputs \perp and adds the queried ciphertext to the list of invalid ciphertexts.

The Ideal-World Oracles: From the definition of IND-CCA security, the decryption oracle must be exactly the same. Thus, only modifications are in the \mathbf{Enc} oracle. The oracle samples random ciphertexts and does not update the lists of valid inputs and outputs.

Algorithm 1 **PRI** and Ideal World Oracles

```

1: Initialize
2:  $\mathcal{L}_c = \{|C| \mid C \in \{0, 1\}^{(s+\lambda)+}\}$ 
3: for  $(T_w, c) \in \mathcal{T} \times \mathcal{L}_c$  do
4:    $\mathcal{M}^{T_w, c} \leftarrow \phi$ 
5:    $\mathcal{C}^{T_w, c} \leftarrow \phi$ 
6:    $\mathcal{I}^{T_w, c} \leftarrow \phi$ 
7:    $\mathcal{V}^{T_w, c} \leftarrow \phi$ 
8:    $\mathcal{O}^{T_w, c} \leftarrow \phi$ 
9: end for
10: bad1  $\leftarrow$  false
11: bad2  $\leftarrow$  false
12: badF  $\leftarrow$  false

1: Enc( $T_w, M$ )
2:  $c \leftarrow |M| + \lambda$ 
3:  $C \xleftarrow{\$} \{0, 1\}^c$ 
4: if  $C \in \mathcal{O}^{T_w, c} \cup \mathcal{C}^{T_w, c} \cup \mathcal{I}^{T_w, c}$  then
5:   bad1  $\leftarrow$  true
6:    $C \xleftarrow{\$} \{0, 1\}^c \setminus$ 
7:    $(\mathcal{O}^{T_w, c} \cup \mathcal{C}^{T_w, c} \cup \mathcal{I}^{T_w, c})$ 
8: end if
9:  $\mathcal{V}^{T_w, c} \leftarrow \mathcal{V}^{T_w, c} \cup \{M\}$ 
10:  $\mathcal{O}^{T_w, c} \leftarrow \mathcal{O}^{T_w, c} \cup \{C\}$ 
11:  $\mathcal{M}^{T_w, c} \leftarrow \mathcal{M}^{T_w, c} \cup \{M\}$ 
12:  $\mathcal{C}^{T_w, c} \leftarrow \mathcal{C}^{T_w, c} \cup \{C\}$ 
13: return  $C$ 

1: Dec( $T_w, C$ )
2:  $M \leftarrow \perp$ 
3:  $c \leftarrow |C|$ 
4:  $m \leftarrow |C| - \lambda$ 
5:  $\mathcal{M}^e \leftarrow \{0, 1\}^m \setminus \mathcal{M}^{T_w, c}$ 
6:  $\mathcal{C}^e \leftarrow \{0, 1\}^c \setminus (\mathcal{C}^{T_w, c} \cup \mathcal{I}^{T_w, c})$ 
7:  $x \xleftarrow{\$} \{1, \dots, |\mathcal{C}^e|\}$ 
8: if  $x \leq |\mathcal{M}^e|$  then
9:   badF  $\leftarrow$  true
10:   $M \xleftarrow{\$} \mathcal{M}^e$ 
11:  if  $M \in \mathcal{V}^{T_w, c}$  then
12:    bad2  $\leftarrow$  true
13:  end if
14:   $\mathcal{M}^{T_w, c} \leftarrow \mathcal{M}^{T_w, c} \cup \{M\}$ 
15:   $\mathcal{C}^{T_w, c} \leftarrow \mathcal{C}^{T_w, c} \cup \{C\}$ 
16: else
17:   $\mathcal{I}^{T_w, c} \leftarrow \mathcal{I}^{T_w, c} \cup \{C\}$ 
18: end if
19: return  $M$ 

```

Bad Events: We define three bad events:

1. **bad1:** This happens if the **Enc** oracle samples randomly a ciphertext that corresponds to a previously assigned point to the random injection, or a ciphertext that have been deemed invalid by the **Dec** oracle. In the real-world, the oracle performs a corrective step, while in the ideal-world such step is not performed.
2. **bad2:** This happens if the **Dec** oracle samples a decrypted message that has been outputted by the **Enc** for the same T_w .
3. **badF:** This is the event that a successful forgery occurs.

Note that **bad2** is impossible in the real world, since all the inputs to **Enc** are automatically excluded from the valid plaintexts. Thus, this event can only happen in the ideal world, when the adversary queries $C \leftarrow \mathbf{Enc}(T_w, M)$, then makes a subsequent query $M' \leftarrow \mathbf{Dec}(T_w, C')$ such that $C' \neq C$ and $M' = M$. On the other hand, **bad1** can happen in both worlds in two ways:

1. **bad1a:** Two queries $C_1 \leftarrow \mathbf{Enc}(T_w, M_1)$ and $C_2 \leftarrow \mathbf{Enc}(T_w, M_2)$, such that $C_1 = C_2$ and $M_1 \neq M_2$.
2. **bad1b:** A query $M_1 \leftarrow \mathbf{Dec}(T_w, C_1)$ is followed by a query $C_2 \leftarrow \mathbf{Enc}(T_w, M_2)$, such that $C_1 = C_2$ and $M_1 \neq M_2$.

Algorithm 2 Intermediate World Oracles

```

1: Initialize
2:  $\mathcal{L}_c = \{|C| \mid C \in \{0, 1\}^{(s+\lambda)+}\}$ 
3: for  $(T_w, c) \in \mathcal{T} \times \mathcal{L}_c$  do
4:    $\mathcal{M}^{T_w, c} \leftarrow \phi$ 
5:    $\mathcal{C}^{T_w, c} \leftarrow \phi$ 
6:    $\mathcal{I}^{T_w, c} \leftarrow \phi$ 
7:    $\mathcal{V}^{T_w, c} \leftarrow \phi$ 
8:    $\mathcal{O}^{T_w, c} \leftarrow \phi$ 
9:    $n^{T_w, c} \leftarrow 0$ 
10: end for
11: bad1  $\leftarrow$  false
12: bad2  $\leftarrow$  false
13: badF  $\leftarrow$  false

1: Enc $(T_w, M)$ 
2:  $c \leftarrow |M| + \lambda$ 
3:  $C \xleftarrow{\$} \{0, 1\}^c$ 
4: if  $C \in \mathcal{O}^{T_w, c} \cup \mathcal{C}^{T_w, c} \cup \mathcal{I}^{T_w, c}$  then
5:   bad1  $\leftarrow$  true
6: end if
7:  $\mathcal{V}^{T_w, c} \leftarrow \mathcal{V}^{T_w, c} \cup \{M\}$ 
8:  $\mathcal{O}^{T_w, c} \leftarrow \mathcal{O}^{T_w, c} \cup \{C\}$ 
9:  $n^{T_w, c} \leftarrow n^{T_w, c} + 1$ 
10: return  $C$ 

1: Dec $(T_w, C)$ 
2:  $M \leftarrow \perp$ 
3:  $c \leftarrow |C|$ 
4:  $m \leftarrow |C| - \lambda$ 
5:  $\mathcal{M}^e \leftarrow \{0, 1\}^m \setminus \mathcal{M}^{T_w, c}$ 
6:  $\mathcal{C}^e \leftarrow \{0, 1\}^c \setminus (\mathcal{C}^{T_w, c} \cup \mathcal{I}^{T_w, c})$ 
7:  $x \xleftarrow{\$} \{1, \dots, |\mathcal{C}^e| - n^{T_w, c}\}$ 
8: if  $x \leq |\mathcal{M}^e| - n^{T_w, c}$  then
9:   badF  $\leftarrow$  true
10:   $M \xleftarrow{\$} \mathcal{M}^e$ 
11:  if  $M \in \mathcal{V}^{T_w, c}$  then
12:    bad2  $\leftarrow$  true
13:  end if
14:   $\mathcal{M}^{T_w, c} \leftarrow \mathcal{M}^{T_w, c} \cup \{M\}$ 
15:   $\mathcal{C}^{T_w, c} \leftarrow \mathcal{C}^{T_w, c} \cup \{C\}$ 
16: else
17:   $\mathcal{I}^{T_w, c} \leftarrow \mathcal{I}^{T_w, c} \cup \{C\}$ 
18: end if
19: return  $M$ 

```

If none of **bad1** or **bad2** occur, then all the queries are compatible with a random injection description. However, the two games are not identical, since the probability distribution of successful forgery is slightly different in different worlds. The real world decryption oracle does not exclude plaintexts and ciphertexts used during encryption queries from the lists of valid plaintexts and ciphertexts. For this purpose, we introduce an intermediate world in Algorithm 2. We shall apply the triangle inequality as follows:

$$\begin{aligned}
& |\Pr[\mathbf{A}^{real} \Rightarrow 1] - \Pr[\mathbf{A}^{ideal} \Rightarrow 1]| \leq |\Pr[\mathbf{A}^{real} \Rightarrow 1] - \Pr[\mathbf{A}^{intermediate} \Rightarrow 1]| + \\
& |\Pr[\mathbf{A}^{intermediate} \Rightarrow 1] - \Pr[\mathbf{A}^{ideal} \Rightarrow 1]|
\end{aligned}$$

Distinguishing the Intermediate and Ideal Worlds: We note that the only difference between the two worlds is in the bias of the coin that determines whether the forgery is successful. Let (T_i, C_i) be the i^{th} decryption query. Let $F_{1,i}$ be the event that the adversary gets a successful forgery in the intermediate world, while $F_{2,i}$ is the event that the adversary gets a successful forgery in the ideal world. Thus, we bound the distinguishing advantage using the statistical distance between the distributions of these events.

$$\begin{aligned}
& |\Pr[\mathbf{A}^{intermediate} \Rightarrow 1] - \Pr[\mathbf{A}^{ideal} \Rightarrow 1]| \leq \\
& \frac{1}{2} \sum_{i=1}^{q_d} |\Pr[F_{1,i} = \mathbf{true}] - \Pr[F_{2,i} = \mathbf{true}]| + \\
& |\Pr[F_{1,i} = \mathbf{false}] - \Pr[F_{2,i} = \mathbf{false}]|
\end{aligned}$$

$$\begin{aligned}
&= \frac{1}{2} \sum_{i=1}^{q_d} |\Pr[F_{1,i} = \text{true}] - \Pr[F_{2,i} = \text{true}]| + \\
&\quad |1 - \Pr[F_{1,i} = \text{true}] - 1 + \Pr[F_{2,i} = \text{true}]| \\
&= \sum_{i=1}^{q_d} |\Pr[F_{1,i} = \text{true}] - \Pr[F_{2,i} = \text{true}]|
\end{aligned}$$

Let q_d^i be the number of decryption queries with the same tweak T_i prior to the i^{th} query, q_f^i is the number of successful forgeries with the same tweak prior to the i^{th} query.

$$\Pr[F_{1,i} = \text{true}] = \frac{2^{m_i} - q_f^i - n^{T_i, c_i}}{2^{c_i} - q_d^i - n^{T_i, c_i}}$$

and

$$\Pr[F_{2,i} = \text{true}] = \frac{2^{m_i} - q_f^i}{2^{c_i} - q_d^i}$$

Thus,

$$\begin{aligned}
&|\Pr[F_{1,i} = \text{true}] - \Pr[F_{2,i} = \text{true}]| = \left| \frac{2^{m_i} - q_f^i}{2^{c_i} - q_d^i} - \frac{2^{m_i} - q_f^i - n^{T_i, c_i}}{2^{c_i} - q_d^i - n^{T_i, c_i}} \right| \\
&= \left| \frac{(2^{c_i} - q_d^i)(2^{m_i} - q_f^i) - n^{T_i, c_i}(2^{c_i} - q_d^i) - (2^{c_i} - q_d^i)(2^{m_i} - q_f^i) + n^{T_i, c_i}(2^{m_i} - q_f^i)}{(2^{c_i} - q_d^i)(2^{c_i} - q_d^i - n^{T_i, c_i})} \right| \\
&= \frac{n^{T_i, c_i}}{(2^{c_i} - q_d^i)(2^{c_i} - q_d^i - n^{T_i, c_i})} |(2^{m_i} - q_f^i) - (2^{c_i} - q_d^i)| \\
&\leq \frac{4\mu}{2^{2c_i}} |(2^{m_i} - q_f^i) - (2^{c_i} - q_d^i)| \leq \frac{4\mu 2^{c_i}}{2^{2c_i}} = \frac{4\mu}{2^{c_i}} \leq \frac{4\mu}{2^{s+\lambda}}.
\end{aligned}$$

Thus,

$$|\Pr[\mathbf{A}^{\text{intermediate}} \Rightarrow 1] - \Pr[\mathbf{A}^{\text{ideal}} \Rightarrow 1]| \leq \frac{4\mu q_d}{2^{s+\lambda}} \quad (1)$$

Distinguishing the Intermediate and Real Worlds The forgery bias in the real and intermediate worlds is the same. Thus, the adversary can only distinguish the two worlds if **bad1** or **bad2** occur. Otherwise, the two worlds are indistinguishable.

Winning Condition: We restrict the game to the case where the game terminates if any of the bad events **bad1** or **bad2** occur. This can only increase the adversary's advantage.

From this description, we know from [Sho04, Lemma 1]

$$|\Pr[\mathbf{A}^{\text{real}} \Rightarrow 1] - \Pr[\mathbf{A}^{\text{intermediate}} \Rightarrow 1]| \leq \Pr[\text{bad1a}] + \Pr[\text{bad1b}] + \Pr[\text{bad2}]$$

bad1a: For a given queries $\mathbf{Enc}(T_i, M_i)$, there are at most $(\mu - 1)$ previous queries with the same T_i . Thus, the probability of the event is bounded by

$$\Pr[\text{bad1a}] \leq \frac{(\mu - 1)q_e}{2^{s+\lambda}} \quad (2)$$

bad1b: Let the i^{th} encryption query be (T_i, M_i) , $c_i = |M_i| + \lambda$ and q_d^i be the number of decryption queries made before the i^{th} encryption query with the same tweak T_i and ciphertext length c_i . Let E_i be the event that **bad1b** is set in the i^{th} encryption query. Let nE_i be the event that **bad1b** is not set in any of the first $i - 1$ encryption queries. Given the game terminates if **bad1b** is set, the probability that **bad1b** is set in any encryption query is

$$\Pr[\text{bad1b}] = \Pr[E_1] + \sum_{i=2}^{q_e} \Pr[E_i | nE_i] \Pr[nE_i] \leq \frac{q_d^1}{2^{c_1}} + \sum_{i=2}^{q_e} \frac{q_d^i}{2^{c_i}} = \sum_{i=1}^{q_e} \frac{q_d^i}{2^{c_i}} \leq \frac{\mu q_d}{2^{s+\lambda}}. \quad (3)$$

The last inequality follows from the worst case when all the encryption queries are performed after all the decryption queries and each tweak appears in at most μ encryption queries.

bad2: For any decryption query with tweak T_j , there are at most μ encryption queries with the same tweak. Let q_d^j be the number of decryption queries with tweak T_j before the j^{th} decryption query and q_f^j is the number of such queries that did not output \perp . For the oracle description, $n^{T_j, c}$ is the number of encryption calls that use the same tweak T_j . In the case of decryption oracle, the probability that **bad2** is set in the j^{th} decryption query in the intermediate world is the probability that the conditions on lines 8 and 10 are set, which is given by

$$\frac{2^{m_j} - q_f^j - n^{T_j, c_j}}{2^{c_j} - q_d^j - n^{T_j, c_j}} \times \frac{1}{2^{m_j} - q_f^j} \leq \frac{1}{2^{c_j} - q_d^j - n^{T_j, c_j}} \leq \frac{2}{2^{s+\lambda}}.$$

Let D_j be the event that **bad2** is set in the j^{th} decryption query, and nD_j be the event that **bad2** is not set in the first $j - 1$ decryption queries, then, given the game terminates if **bad2** is set, the probability that **bad** is set during the first q_d decryption queries is given by

$$\begin{aligned} \Pr[D_1] + \sum_{j=2}^{q_d} \Pr[D_j | nD_j] \Pr[nD_j] &\leq \sum_{j=1}^{q_d} \frac{2}{2^{s+\lambda}} \\ &= \frac{2q_d}{2^{s+\lambda}}. \end{aligned} \quad (4)$$

The overall bound follows from Equations 1, 2, 3 and 4. \square

This result gives the first ingredient of building schemes with IND-CCA security with a fixed stretch; set a minimum plaintext length. However, we have to be careful when using this result. One tempting approach to bound the IND-CCA security is to first bound the PRI security, then rely on Theorem 2 to get the final bound. However, this approach may lead to loose bounds. In the IND-CCA game, we allow the decryption oracle to be as weak as the scheme itself. This means that forged messages may exhibit non-random patterns in the real world, without the scheme being considered insecure. However, by making the transition to PRI first, we penalize the scheme for potentially having such non-random patterns in forged messages. Thus, for a given scheme, it is still useful to perform a dedicated IND-CCA analysis.

3.1 Matching Attack on the Minimum Plaintext Length

The result from Theorem 2 presents somewhat bad news for instantiating AEAD in general, since the standard AEAD syntax allows for messages to be short and even empty strings. In this section, we show that, unfortunately, this dependence on the minimum plaintext length is tight, by giving a nonce-respecting adversary that breaks IND-CCA security with $O(2^{s+\lambda})$ queries.

Theorem 3. For any AEAD scheme Π with plaintext domain $\{0, 1\}^{s+}$ and a fixed stretch λ , there is a nonce-respecting IND-CCA adversary \mathbf{A} such that

$$\text{Adv}_{\Pi}^{\text{ind-cca}}(\mathbf{A}) > \frac{0.5q_d}{2^{s+\lambda}}$$

where q_d is the number of decryption oracle queries. In the special case where the plaintext space is $\{0, 1\}^*$, which includes the empty string,

$$\text{Adv}_{\Pi}^{\text{ind-cca}}(\mathbf{A}) > \frac{0.5q_d}{2^{\lambda}}.$$

Proof. We construct an adversary \mathbf{A} as follows:

1. \mathbf{A} selects a random ciphertext C' , such that $|C'| = s + \lambda$.
2. \mathbf{A} selects a random tweak T_w and asks for the decryption $\text{Dec}(T_w, C') \Rightarrow M'$.
3. \mathbf{A} selects a random message M , such that $|M| = s$ and $M \neq M'$, and asks for the encryption $\text{Enc}(T_w, M)$.
4. \mathbf{A} terminates and returns 1 if $C = C'$.

\mathbf{A} repeats this attack q_d times, with a different T_w each time. In the real-world, the condition in step 4 can never occur. In the ideal-world, since \mathbf{A} selected both C' and M randomly, $\Pr[C = C' | M \neq M'] = 1/2^{s+\lambda}$. Since \mathbf{A} performs q_d independent attempts,

$$\text{Adv}_{\Pi}^{\text{ind-cca}}(\mathbf{A}) = 1 - \left(1 - \frac{1}{2^{s+\lambda}}\right)^{q_d} > \frac{0.5q_d}{2^{s+\lambda}}.$$

□

4 Insufficiency of MRAE security

An MRAE scheme is a scheme that allows the adversary to repeat the nonce while maintaining AEAD security. If the nonce is treated as a constant, the scheme becomes a DAE scheme. While [RS06] showed that a DAE scheme and a PRI are indistinguishable, PRIs can be quite expensive, and DAE/MRAE can be achieved using cheaper methods. In this section, we give a few examples of schemes that achieve MRAE security, while having their IND-CCA attacks with 2^{λ} queries.

4.1 Generalized nonce-based synthetic IV schemes

In this section, we study the SIV structure in different variations, and generalize it to be nonce-based. This generalization is not new, as it was proposed in [PS16] and used in the SCT and Romulus-M [IKMP20] modes. We start by defining nonce-IV-based encryption, then consider the different instances.

Definition 3. A nonce-IV-based encryption scheme is a tuple $\Pi(\mathcal{K}, \mathcal{E}, \mathcal{D})$, where $\mathcal{E} : \mathcal{K} \times \mathcal{N} \times \mathcal{A} \times \mathcal{IV} \times \{0, 1\}^{s+} \rightarrow \mathcal{IV} \times \{0, 1\}^{s+}$ is a keyed encryption function with key space \mathcal{K} , plaintext/ciphertext space $\{0, 1\}^{s+}$, nonce space \mathcal{N} , header space \mathcal{A} and IV space \mathcal{IV} , while $\mathcal{D} : \mathcal{K} \times \mathcal{N} \times \mathcal{A} \times \mathcal{IV} \times \{0, 1\}^{s+} \rightarrow \mathcal{IV} \times \{0, 1\}^{s+}$ is its decryption function. $\mathcal{E}(K, N, A, \cdot, \cdot)$ is a permutation for its fourth and fifth inputs, and $\mathcal{D}(K, N, A, \mathcal{E}(K, N, A, IV, M)) = (IV, M)$.

Definition 4. Let $\Pi(\mathcal{K}, \mathcal{E}, \mathcal{D})$ be a nonce-IV-based encryption scheme. Let \mathbf{A} be an IND-CCA adversary against Π . In the real world, \mathbf{A} interacts with \mathcal{E}_K and \mathcal{D}_K , while in the ideal world, interacts with $\$$ and \mathcal{D}_K . $\$$ behaves the same as \mathcal{E}_K except that it replaces

the ciphertext generated in each encryption query by a random string of the same length. Let $\text{Adv}_{\Pi}^{\text{ind-cca}}(\mathbf{A})$ denote the IND-CCA advantage of Π against \mathbf{A} . It is defined as

$$\text{Adv}_{\Pi}^{\text{ind-cca}}(\mathbf{A}) \stackrel{\text{def}}{=} |\Pr[K \xleftarrow{\$} \mathcal{K} : \mathbf{A}^{\mathcal{E}_K, \mathcal{D}_K} \Rightarrow 1] - \Pr[K \xleftarrow{\$} \mathcal{K} : \mathbf{A}^{\$, \mathcal{D}_K} \Rightarrow 1]|$$

Note that the definition of IND-CCA security here differs from the definition of sTPRP security, in that the sTPRP security compares a scheme to a family of random permutation, while IND-CCA compares it to a random oracle. Besides, sTPRP security assumes that, in the ideal world, the decryption oracle is ideal and outputs random strings (up to the being a permutation), while IND-CCA security, as defined above, assumes the decryption is always real and can have weaknesses.

Definition 5. Let $f : \mathcal{K}_1 \times \mathcal{N} \times \mathcal{A} \times \{0, 1\}^{s+} \rightarrow \{0, 1\}^\lambda$ be a PRF, and $\Pi(\mathcal{K}_2, \mathcal{E}, \mathcal{D})$ be a nonce-IV-based encryption scheme, where $\mathcal{E} : \mathcal{K}_2 \times \mathcal{N} \times \mathcal{A} \times \{0, 1\}^\lambda \times \{0, 1\}^{s+} \rightarrow \{0, 1\}^\lambda \times \{0, 1\}^{s+}$ is a keyed encryption function with key space \mathcal{K}_2 , plaintext/ciphertext space $\{0, 1\}^{s+}$, nonce space \mathcal{N} , header space \mathcal{A} and IV space \mathcal{TV} , while $\mathcal{D} : \mathcal{K}_2 \times \{0, 1\}^\lambda \times \mathcal{N} \times \mathcal{A} \times \{0, 1\}^{s+} \rightarrow \{0, 1\}^\lambda \times \{0, 1\}^{s+}$ is its decryption function. Then the nonce-based Synthetic IV (nSIV) scheme is a tuple $(\mathcal{K}, \text{Enc}, \text{Dec})$, described in Figure 3.

Algorithm 3 Algorithmic description of the nSIV scheme

1: $\text{nSIV.Enc}(K, N, A, M)$ 2: $K_1 \leftarrow K[1]$ 3: $K_2 \leftarrow K[2]$ 4: $IV \leftarrow f(K_1, N, A, M)$ 5: $(T, C) \leftarrow \mathcal{E}(K_2, N, A, IV, M)$ 6: return (T, C)	1: $\text{nSIV.Dec}(K, N, A, T, C)$ 2: $K_1 \leftarrow K[1]$ 3: $K_2 \leftarrow K[2]$ 4: $(IV^*, M) \leftarrow \mathcal{D}(K_2, N, A, T, C)$ 5: $IV' \leftarrow f(K_1, N, A, M)$ 6: if $IV' = IV^*$ then 7: return M 8: else 9: return \perp 10: end if
---	--

Matching attacks on nSIV with IND-CCA-insecure encryption nSIV achieves MRAE security even if the underlying encryption scheme is not IND-CCA secure. An interesting question is whether nSIV can achieve IND-CCA security beyond the tag length. This is also motivated by the results of [Kha22] and [HII⁺22], which show that online AE cannot achieve such security. We now show that if the underlying encryption scheme of nSIV is not IND-CCA secure, then it cannot achieve the required security level. In this section, we discuss two examples. First, we attack SIV from [RS06] where \tilde{E} is online. Second, we discuss how to apply this attack to SIV when \tilde{E} is a stream-cipher-like encryption scheme *e.g.*, counter mode. In Section 5, we discuss a recently proposed scheme in CRYPTO 2022, namely UIV [DK22] and apply a variant of the second attack to it.

Definition 6. Let e be a nonce-IV-based encryption scheme. We say e is online if $\exists n \in \mathbf{Z}^+$, *s.t.* e satisfies that for two queries $C = e.\mathcal{E}_K^N(IV, M)$ and $C' = e.\mathcal{E}_K^N(IV, M')$,

$$\lfloor M \rfloor_n = \lfloor M' \rfloor_n \iff \lfloor C \rfloor_n = \lfloor C' \rfloor_n$$

Theorem 4. Let $\text{nSIV}[f, e]$ be the AEAD encryption scheme given in depicted in Figure 1. There is an IND-CCA adversary \mathbf{A} against $\text{nSIV}[f, e]$ that makes 1 encryption query and q_d decryption queries such that

$$\text{Adv}_{\text{nSIV}[f, e]}^{\text{ind-cca}}(\mathbf{A}) \geq \frac{0.5q_d}{2^\lambda} \left(1 - \frac{1}{2^n}\right).$$

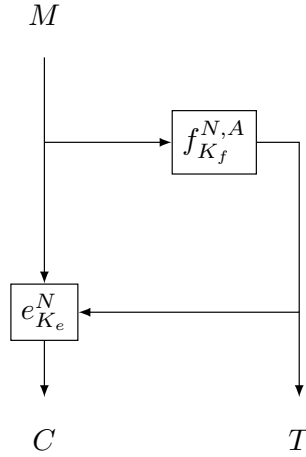


Figure 1: The nSIV scheme with an online encryption scheme

Proof. We construct \mathbf{A} as follows:

1. \mathbf{A} asks for an encryption query $(T, C) = \mathbf{Enc}(N, A, M)$ such that $|M| = \max(2n, 2\lambda)$.
2. \mathbf{A} selects $C^* \xleftarrow{\$} \{0, 1\}^{|M|-n}$ and sets $C' = [C]_n \| C^*$.
3. \mathbf{A} asks for a decryption query $M' = \mathbf{Dec}(N, A, T, C')$.
4. If $M' = \perp$, repeats steps 2 and 3.
5. If $M' \neq \perp$ and $[M']_n = [M]_n$, return 0, otherwise return 1.

After each decryption query, the probability of step 5 getting executed is $1/2^\lambda$. After q_d decryption query, the probability that \mathbf{A} executes step 5 is lower bounded by $0.5q_d/2^n$. If step 5 gets executed in the real world, the condition is always satisfied, while it is satisfied in the ideal world with probability $1/2^n$. \square

The adversary described above breaks the IND-CCA game with about 2^λ decryption queries. However, if e is a stream-cipher-like encryption scheme, such as the one in Figure 2, the adversary can be adapted to act as a decryption oracle. The adversary receives a ciphertext corresponding an unknown message. Then, the adversary performs steps 2-4, with $n = 0$. Once it gets $M' \neq \perp$, it decrypts $M = M' \oplus C' \oplus C$. This attack also shows a matching attack on the Encode-then-Encipher scheme based on UIV proposed in [DK22]. We study matching attacks on plain AEAD schemes proposed in [DK22] in Section 5.

5 Matching Attacks on Rugged PRPs

In CRYPTO 2022, Degabriele and Karadžić [DK22] proposed the concept of rugged PRPs as a new security notion that helps achieve some of the security goals required from an Encode-then-Encipher scheme, without the underlying enciphering scheme being a strong TPRP, which sounds like a similar problem to the problem we are studying. They proposed an underlying encryption scheme called UIV and proposed two AEAD modes based on this scheme. The first is the normal Encode-then-Encipher scheme. The second is an Encode-then-Decipher scheme, where the AEAD encryption uses the decryption algorithm of the underlying encryption scheme and vice-versa. In the former case, the redundancy bits can be set to a constant, while in the latter case, the redundancy bits are set by hashing

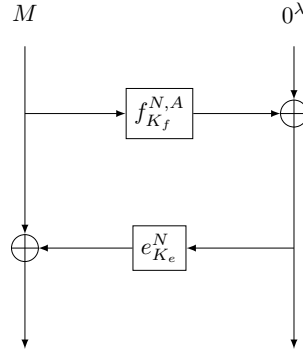


Figure 2: The nSIV scheme with an stream-cipher-like encryption scheme

the nonce, associated data and plaintext using a collision-resistant hash function. In this section, we give a description of the UIV encryption scheme, then we show a matching attack on its Encode-then-Encipher mode that is a variant of the attack in Theorem 4, and two matching attacks on its Encode-then-Decipher mode; one on the original mode and another for the strengthened case where the key stream generation PRF takes the nonce as an auxiliary input. These constructions are depicted in Figure 3.

We would like to highlight that the rugged PRP study and our study share a similar philosophy; can we design AEAD schemes that have security similar to that of an Encode-then-Encipher mode instantiated with a strong TPRP using weaker primitives? However, the two studies deviate in terms of what are the security goals they actually try to achieve. Both studies target IND-CCA security, but [DK22] targets IND-CCA security when the stretch is sufficiently large and (in the case of Encode-then-Decipher) unverified plaintexts are leaked. We, on the other hand, focus on the IND-CCA security when the stretch is not long enough for the required security. This section serves to show that rugged PRPs are not sufficient to achieve our goal.

Unilaterally-Protected IV (UIV) The scheme uses a fixed n -bit block length and variable tweak length TBC and a variable output length, fixed input length PRF. It is depicted in Figure 3 (a), where X is treated as the IV of the scheme. Informally, the goal of the scheme is to behave as a PRP over X , while being IND-CPA-secure over M . However, we observe that if X is treated as part of the nonce, or fixed to a constant, then the scheme can be described using the SIV paradigm. This is exactly the case of the Encode-then-Encipher mode. We can adapt the attack from Theorem 4 to attack the Encode-then-Encipher mode. The adversary operates the same way until the decrypted message (when used as tweak to the TBC) maps T to 0^n . In this case, $\lambda = n$ and the analysis of the attack is the same.

Encode-then-Decipher from UIV The authors of [DK22] propose using UIV in the reverse direction in the Encode-then-Decipher mode shown in Figure 3 (c). In this case, $X = H(N, A, M)$ where H is a deterministic collision-resistant hash function, and $\lambda = n = |H(N, A, M)|$. This already shows the issue, where to get the security of the scheme depends on the collision-resistance of H , which limits the security to at most $\lambda/2$. We show that even for a nonce-respecting adversary, if it runs in time $\geq 2^{\lambda/2}$, the security is tight. The adversary \mathbf{A} operates as follows:

1. \mathbf{A} finds a collision $H(N_1, A_1, M_1) = H(N_2, A_2, M_2)$, ensuring $N_1 \neq N_2$ and $|M_1| = |M_2| = 2\lambda$.
2. \mathbf{A} asks for the encryption query $(T_1, C_1) = \mathbf{Enc}(N_1, A_1, M_1)$.

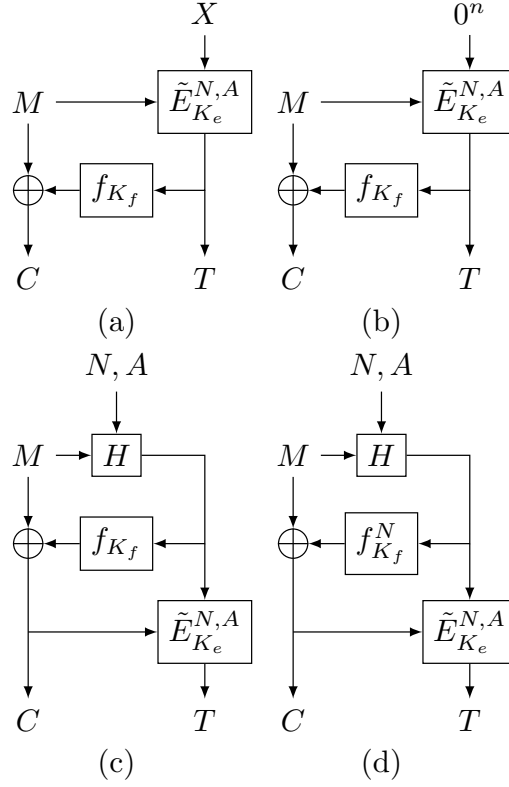


Figure 3: The constructions from [DK22]. (a) The UIV encryption scheme. (b) The UIV-based Encode-then-Encipher scheme. (c) The UIV-based Encode-then-Decipher scheme. (d) The UIV-based Encode-then-Decipher scheme strengthened with nonce-based keystream generation.

3. **A** asks for the encryption query $(T_2, C_2) = \mathbf{Enc}(N_2, A_2, M_2)$.
4. If $M_1 \oplus M_2 = C_1 \oplus C_2$, **A** returns 0, otherwise, it returns 1.

In the real world, this happens with probability 1, while in the ideal world this happens with probability $2^{-2\lambda}$. **A** makes only two encryption queries and runs in roughly the time it needs to find the hash collision. For a practical hash function, after about $2^{\lambda/2}$ hash attempts, the collision can be found with high probability.

Strengthened Encode-then-Decipher from UIV In Figure 3 (d), we envision a slightly strengthened version, where the nonce is also included as an input to f . This prevents the previous attack. However, we show that while this makes the attack qualitatively harder, it does not affect the security level. We consider an adversary **B** that operates as follows:

1. **B** finds a collision $H(N, A_1, M_1) = H(N, A_2, M_2)$, and $|M_1| = |M_2| = 2\lambda$.
2. **B** asks for the encryption query $(T_1, C_1) = \mathbf{Enc}(N, A_1, M_1)$.
3. **B** sets $C_2 = C_1 \oplus M_1 \oplus M_2$.
4. **B** samples T_{\S} from $\{0, 1\}^\lambda$ without replacement.
5. **B** asks for the encryption query $M' = \mathbf{Dec}(N, A_2, T_{\S}, C_2)$.

6. If $M' = M_2$, it returns 0, otherwise, it repeats steps 4 and 5.

In the ideal world, \mathbf{B} can only return 0 with probability approximately $q_d/2^{3\lambda}$, where the forgery has to succeed and $M' = M_2$ has to happen when C_1 is selected randomly. In the real world, since T is generated using a permutation, there must exist a value T_{\S} where $H(N, A_2, M_2) = (\bar{E}^{-1})_{K_e}^{N, A_2}(T_{\S})$. Hence, by trying all possible values until such value is found, \mathbf{B} always returns 1 after at most $q_d = 2^\lambda$. \mathbf{B} runs in the time needed to find the collision plus the time needed to perform 2^λ decryption queries and 1 encryption query.

6 Conclusions

In this paper, we studied some of the gaps in the area of IND-CCA security of AEAD schemes. We expanded on the results of [Kha22] and [HHI⁺22], which dealt with online AE and fixed-length Encode-then-Encipher schemes. We first show that PRI security implies IND-CCA up to the minimum ciphertext length. This generalizes and tightens the result of [HHI⁺22], showing that (2) an enciphering scheme is not needed, but is close to optimal, and (b) IND-CCA security beyond the tag size with a fixed tag length is impossible unless the plaintext space is restricted with a minimum size. Then, we show a matching attack that breaks any AEAD scheme with $O(2^{s+\lambda})$ queries where s is the minimum plaintext size and λ is the tag size. On a different note, we show that two-pass schemes, such as SIV and rugged PRP cannot achieve IND-CCA security beyond the tag length.

Acknowledgements

I would like to thank the anonymous reviewers for their insightful comments. I would like to also thank Nicky Mouha and Kazuhiko Minematsu for independently pointing to rugged PRPs as a potential target for short tags. This work was partly done in the Seagate Research Group, Singapore. The author is supported by the Wallenberg-NTU Presidential Postdoctoral Fellowship.

References

- [AFL⁺16] Farzaneh Abed, Christian Forler, Eik List, Stefan Lucks, and Jakob Wenzel. RIV for robust authenticated encryption. In *Fast Software Encryption: 23rd International Conference, FSE 2016*, pages 23–42. Springer, 2016. doi:10.1007/978-3-662-52993-5_2.
- [BN00] Mihir Bellare and Chanathip Namprempre. Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. In Tatsuaki Okamoto, editor, *Advances in Cryptology — ASIACRYPT 2000*, pages 531–545. Springer Berlin Heidelberg, 2000. doi:10.1007/3-540-44448-3_41.
- [CG16] Colin Chaigneau and Henri Gilbert. Is AEZ v4. 1 sufficiently resilient against key-recovery attacks? *IACR Transactions on Symmetric Cryptology*, 1:654–682, 2016. doi:10.13154/tosc.v2016.i1.114-133.
- [CIMN20] Avik Chakraborti, Tetsu Iwata, Kazuhiko Minematsu, and Mridul Nandi. Blockcipher-based Authenticated Encryption: How Small Can We Go? *Journal of Cryptology*, 33(3):703–741, 2020. doi:10.1007/978-3-319-66787-4_14.
- [com19] Crypto competitions. Caesar: Competition for authenticated encryption: Security, applicability, and robustness, Feb 2019. URL: <https://competitions.cr.yp.to/caesar.html>.

- [DK22] Jean Paul Degabriele and Vukašin Karadžić. Overloading the nonce: rugged PRPs, nonce-set AEAD, and order-resilient channels. In *Advances in Cryptology–CRYPTO 2022: 42nd Annual International Cryptology Conference, CRYPTO 2022, Santa Barbara, CA, USA, August 15–18, 2022, Proceedings, Part IV*, pages 264–295. Springer, 2022. doi:10.1007/978-3-031-15985-5_10.
- [HII⁺22] Akinori Hosoyamada, Akiko Inoue, Ryoma Ito, Tetsu Iwata, Kazuhiko Mimematsu, Ferdinand Sibleyras, and Yosuke Todo. Cryptanalysis of Rocca and Feasibility of Its Security Claim. *IACR Transactions on Symmetric Cryptology*, 2022(3):123–151, Sep. 2022. URL: <https://tosc.iacr.org/index.php/ToSC/article/view/9852>, doi:10.46586/tosc.v2022.i3.123-151.
- [HKR15] Viet Tung Hoang, Ted Krovetz, and Phillip Rogaway. Robust Authenticated-Encryption AEZ and the Problem That It Solves. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology – EUROCRYPT 2015*, pages 15–44, Berlin, Heidelberg, 2015. Springer Berlin Heidelberg. doi:10.1007/978-3-662-46800-5_2.
- [IKMP20] Tetsu Iwata, Mustafa Khairallah, Kazuhiko Minematsu, and Thomas Peyrin. Duel of the Titans: the Romulus and Remus Families of Lightweight AEAD Algorithms. *IACR Transactions on Symmetric Cryptology*, pages 43–120, 2020. doi:10.13154/tosc.v2020.i1.43-120.
- [Kha22] Mustafa Khairallah. Security of COFB against chosen ciphertext attacks. *IACR Transactions on Symmetric Cryptology*, pages 138–157, 2022. doi:10.46586/tosc.v2022.i1.138-157.
- [oST19] National Institute of Standardization and Technology. Lightweight cryptography, 2019. URL: <https://csrc.nist.gov/Projects/lightweight-cryptography>.
- [PS16] Thomas Peyrin and Yannick Seurin. Counter-in-Tweak: Authenticated Encryption Modes for Tweakable Block Ciphers. In *Advances in Cryptology–CRYPTO 2016: 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part I*, pages 33–63. Springer, 2016. doi:10.1007/978-3-662-53018-4_2.
- [Rog02] Phillip Rogaway. Authenticated-Encryption with Associated Data. In *Proceedings of the 9th ACM Conference on Computer and Communications Security*, pages 98–107, 2002. doi:10.1145/586110.586125.
- [RS06] Phillip Rogaway and Thomas Shrimpton. A Provable-Security Treatment of the Key-Wrap Problem. In Serge Vaudenay, editor, *Advances in Cryptology - EUROCRYPT 2006*, pages 373–390, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg. doi:10.1007/11761679_23.
- [Sho04] Victor Shoup. Sequences of games: a tool for taming complexity in security proofs. Cryptology ePrint Archive, Paper 2004/332, 2004. <https://eprint.iacr.org/2004/332>. URL: <https://eprint.iacr.org/2004/332>.
- [SLN⁺21] Kosei Sakamoto, Fukang Liu, Yuto Nakano, Shinsaku Kiyomoto, and Takanori Isobe. Rocca: an efficient AES-based encryption scheme for beyond 5G. *IACR Transactions on Symmetric Cryptology*, pages 1–30, 2021. doi:10.46586/tosc.v2021.i2.1-30.