CCM-SIV: Single-PRF Nonce-Misuse-Resistant Authenticated Encryption

Patrick Kresmer and Alexander Zeh

Research and Development Center
Infineon Technologies AG, Munich, Germany
{patrick.kresmer, alexander.zeh}@infineon.com

Abstract. We propose a new nonce-misuse-resistant authenticated encryption scheme, which instantiates the SIV paradigm of Rogaway and Shrimpton. In contrast to the GCM-SIV approach proposed by Gueron and Lindell, we do only use a single type of cryptographic primitive, which can be advantageous in restricted embedded devices. Furthermore, we use three independent and fixed subkeys derived from a single master key. Similar to the CCM mode, our scheme uses a combination of the CTR mode for the symmetric encryption and a MAC based on the CBC construction and is therefore called CCM-SIV. We provide a detailed security proof for our scheme. Furthermore, we outline its extension to a nonce-based key derivation as the AES-GCM-SIV approach.

Keywords: AEAD, AES-GCM, AES-GCM-SIV, AES-CCM, Nonce

1 Introduction

To efficiently achieve authenticity, integrity and confidentiality on the communication channels, authenticated encryption with associated data (AEAD) schemes are used. A famous and widely deployed AEAD system is the Galois-Counter Mode-of-Operation of the AES block cipher (AES-GCM, [15]). The security of the scheme heavily relies on the uniqueness of the nonces, since they are used as a "randomizing" input for the AES-GCM encryption. If they happen to be not unique, the security of AES-GCM is completely broken, since the plaintext and secret hashing keys can be recovered (see, e.g., [2, 12]).

GCM-SIV [7] is based on the same building blocks as AES-GCM, but it combines them through the SIV composition method [19]. The SIV paradigm endows the resulting scheme with "nonce-misuse-resistance". A nonce-based key derivation for GCM-SIV [7] was proposed by Gueron *et al.* [14] (and denoted as AES-GCM-SIV) and by Iwata and Minematsu in [10]. The original security bounds of GCM-SIV were re-considered by Iwata and Seurin [11] and led to an update of [6] in July 2017 and [8].

AES Counter with CBC-MAC (AES-CCM, [4]) is, as AES-GCM, a specified cipher suite for IPsec [9], TLS [1] and relies also on the uniqueness of the nonces. AES-CCM combines the CBC-MAC and the well-known counter (CTR) mode and uses a single type of PRF (in contrast to AES-GCM where a second function for the Galois field multiplication is required). Clearly, operations of AES-GCM can be parallelized and, e.g., Intel's PCLMULQDQ instruction [5] allows impressive fast realizations.

Our goal is to deploy nonce-misuse resistant AEAD schemes in restricted embedded systems with fault-tolerant requirements [17], where operating elements are often realized dual (or triple) modular redundant. In this context, it is beneficial that AES-CCM (resp. CCM-SIV) does not require a second cryptographic primitive. A reference implementation of CCM-SIV can be found on gitlab [13].

Contribution. We propose a new nonce-misuse-resistant authenticated encryption scheme called CCM-SIV based on the CTR mode of operation and a dedicated CBC-based MAC. Our scheme only uses a single type of cryptographic primitive as the main building block and derives three independent subkeys from a single masterkey by a deterministic pseudo random number generator (PRNG). We provide a detailed security proof of our scheme. The following main theorem summarizes our result:

Theorem 1. *[Security of CCM-SIV.] Let $\widetilde{\Pi}$ denote the nonce-misuse-resistant authenticated encryption (nAE) scheme CCM-SIV defined by Algorithm 8 and $\mathcal{A}$ be a probabilistic polynomial time (PPT) adversary, who wins the* nAE *game defined in Figure 11 with non-negligible advantage. Then there exists a wrapping PPT adversary $\mathcal{B}$ playing the* PRF *game, who calls $\mathcal{A}$ as a subroutine and is able to distinguish the underlying PRF E from a truly random function with non-negligible advantage. Particular, it holds that*

$$\mathbf{adv}^{\mathrm{nAE}}_{\widetilde{\Pi}}(\mathcal{A}) \leq 2 \cdot \mathbf{adv}^{\mathrm{PRF}}_{E}(\mathcal{B}) + \epsilon \cdot \frac{(q_e + q_d)^2}{2} + \frac{q_e^2 \cdot p_{\max}}{2 \cdot 2^x} + \frac{q_d}{2^x},$$

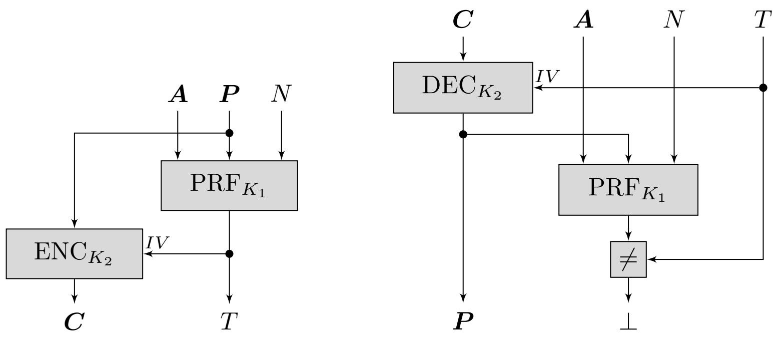
where ϵ is the security bound of the underlying ϵ-almost XOR universal hash function family used in the tag generation, q_e the number of the (polynomially bounded) encryption queries of $\mathcal{A}$, q_d the number of the (polynomially bounded) decryption queries of $\mathcal{A}$ and x the block size of E. In the PRF *security game, $\mathcal{B}$ sends his encryption queries to the oracle, which is either the PRF E_K with $K \xleftarrow{\$} \mathcal{K}$ or a truly random function.*

In Section 2 we introduce the notation used throughout this paper and we recall the SIV paradigm. Section 3 provides the reader with a detailed description of the CCM-SIV key generation, authenticated encryption and authenticated decryption. The security analysis of our new nonce-misuse resistant AEAD scheme CCM-SIV represents the main part of this paper and is done in four steps in Section 4. Section 6 gives a short summary and conclusion.

2 Notation

For a number $x \in \mathbb{N}^+$, the set $\{0,1\}^x$ denotes the set of all bit strings of size x and $\{0,1\}^*$ denotes the countable set of all possible bit strings. The script letters $\mathcal{A}, \mathcal{B}, \mathcal{D}$ are used for algorithms, whereas all other script letters such as $\mathcal{X}$ are used to denote sets, e.g. $\mathcal{X} = \{0,1\}^x$. Upper-case letters such as X are used for a single data block, for example $X \in \mathcal{X}$, where $\mathcal{X} = \{0,1\}^x$ and x is the block size. Upper-case letters such as E, F, H are used for functions. E_K denotes the AES block cipher under key K. Messages, that consist of multiple blocks are denoted with bold letters, e.g. $\boldsymbol{X} = (X_1, X_2, \dots, X_m)$ for an m-block message, where each $X_1, X_2, \dots X_m \in \mathcal{X}$ and $\boldsymbol{X} \in \mathcal{X}^m$. Bold letters are also used for elements in $\{0,1\}^*$. The symbol $\mathcal{X}^{\leq m_{\max}}$ denotes the set of all messages, that consists of at most $m_{\max}$ data blocks. Lower-case letters are used for different purposes such as indices, size values or functions. For a data block $X \in \mathcal{X}$ and $\mathcal{X} = \{0,1\}^x$, $|X|$ denotes the bit-size x of X, whereas $|\mathcal{X}|$ denotes the number of all possible values for X (i.e. $|\mathcal{X}| = 2^x$). The vector symbol $\vec{X}$ is used for vector-valued messages, which may contain several different elements of different lengths, e.g. $\vec{X} = (\boldsymbol{A}, \boldsymbol{B}, C)$ with $\boldsymbol{A}, \boldsymbol{B} \in \{0,1\}^*$ and $C \in \mathcal{X}$. The concatenation of two bit strings $\boldsymbol{X}, \boldsymbol{Y} \in \{0,1\}^*$ is written as $\boldsymbol{X} \parallel \boldsymbol{Y}$. $\Pr[\text{event} \mid \text{cond}]$ denotes the probability that event occurs conditioned to the case, that cond has occurred. $X \xleftarrow{\$} \mathcal{X}$ means, that X is chosen uniformly and independently from the set $\mathcal{X}$. We use "truly random" as a synonym for uniformly distributed and independent values. $\text{Funs}_{a,b\to c}$ denotes the set of all possible functions mapping $\{0,1\}^a \times \{0,1\}^b \to \{0,1\}^c$. The adversary's advantage $\mathbf{adv}_{\Pi}^{\mathrm{G}}(\mathcal{A})$ denotes the absolute difference of the probability that an adversary $\mathcal{A}$ wins a security game G and the probability that an adversary $\mathcal{A}$ loses a security game G. In the security game G, the adversary $\mathcal{A}$ communicates with an oracle $\mathcal{O}$, that is either the scheme Π itself (pseudo-random oracle) or its idealized mathematical model (truly random oracle). Winning the game means, the adversary guesses the correct oracle version. $\Pr\left[\mathcal{A}^F \Rightarrow \mathtt{PR}\right]$ denotes the probability, that an adversary $\mathcal{A}$ guesses "pseudo-random" after talking to a function F, whereas $\Pr\left[\mathcal{A}^{\text{Funs}_{x\to y}} \Rightarrow \mathtt{R}\right]$ denotes the probability, that she guesses "random" after talking to a uniformly chosen function out of the set $\text{Funs}_{x\to y}$.

The SIV construction [18], respectively construction A4 of Namprempre *et al.* [16] is a general paradigm of combining a secure IV-based symmetric encryption scheme with a vector-input PRF to obtain a secure nonce-based and nonce-misuse-resistant authenticated encryption scheme. Both primitives have two distinct and independent keys K_1 and K_2. The main idea is to use the authentication tag T as the IV for the IV-based symmetric encryption. Since the MAC consists of a PRF, its output is uniformly and independently distributed as long as the inputs to the PRF are all distinct. This means, the tag looks like a randomly chosen value and hence it meets the requirements for an IV. Figure 1(a) and 1(b) illustrate the SIV encryption and decryption. For the calculation of the tag T, the inputs for the PRF are the additional data $\boldsymbol{A}$, the plaintext $\boldsymbol{P}$ and the nonce N. Clearly, $\boldsymbol{A}$ and $\boldsymbol{P}$ must be inputs to the PRF, because we want to provide authenticity/integrity for both. The nonce must also be an input to the PRF. Even if the same message $(\boldsymbol{A}, \boldsymbol{P})$ is sent twice, the ciphertext-tag pair must be distinct. This obviously cannot be the case, if the nonce requirement is violated. Since the PRF and the symmetric encryption scheme are deterministic functions, if the same triple $(\boldsymbol{A}, \boldsymbol{P}, N)$ is encrypted twice, then the tag, the IV



(a) SIV Encryption (b) SIV Decryption

Fig. 1: *For the encryption (a), first, the tag T is calculated with a vector-input PRF over the additional data $\boldsymbol{A}$, the plaintext $\boldsymbol{P}$ and the nonce N, and then the tag is used as the synthetic IV for the symmetric encryption of the plaintext. The ciphertext $\boldsymbol{C}$ is decrypted (b) by using the received tag T as the synthetic IV and then the vector-input PRF is evaluated over the additional data, the resulting plaintext and the nonce. If both tags are not equal, an invalidity symbol $\perp$ is set.*

and the ciphertext are also the same. Consequently, when the same nonce is used twice, then an attacker learns, if the same message was encrypted or not, but nothing beyond that.

An inherent drawback of this construction is, that the symmetric encryption has to wait for the IV, before it can start the encryption. The SIV encryption is inherently sequential: First, the PRF has to be computed and after that, the encryption of the plaintext can start. This means, the plaintext has to be temporarily stored somewhere or loaded twice. During the SIV decryption (Figure 1(b)), the decryption of the ciphertext $\boldsymbol{C}$ can immediately start. The required IV is the received tag T. As soon as on block of ciphertext is decrypted, the corresponding plaintext block is fed into the PRF. The output of the PRF is compared to the received tag T. If they are not equal, the invalidity symbol $\perp$ is outputted instead of the plaintext.

3 CCM-SIV

The name "CCM-SIV" suggests, that the scheme is a straightforward SIV-variant of the CCM mode-of-operation [20]. However, the only building blocks that CCM-SIV has in common with standard CCM is, that it also uses a combination of CTR mode for symmetric encryption and a MAC based on the CBC construction. The CCM-SIV scheme, which is proven to be a secure nonce-misuse-resistant authenticated encryption scheme in Section 4, is shown in Figure 2 and 3. It consists of a combination of a special PRF based on CBC and the CTR mode for the IV-based encryption/decryption. The CCM-SIV authen-

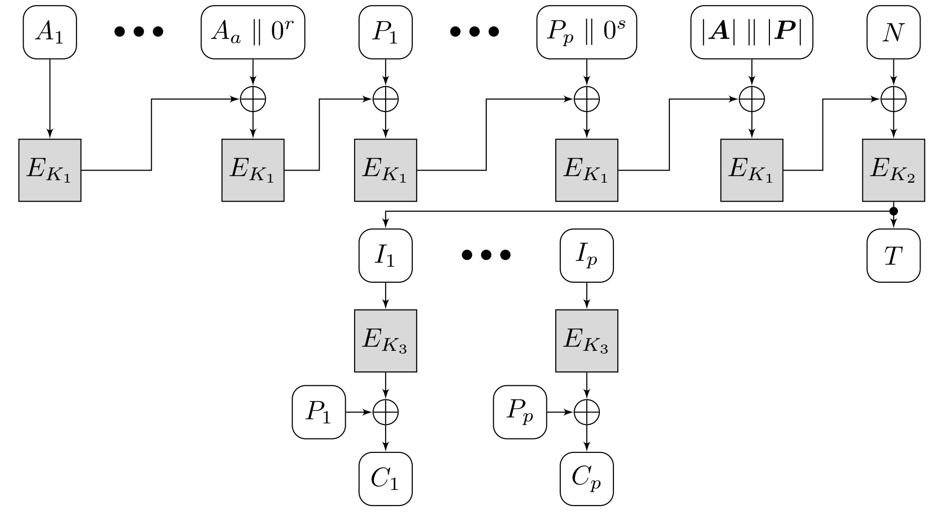


Fig. 2: *CCM-SIV encryption. The tag T is calculated using a dedicated vector-input PRF based on the CBC construction over an encoded input consisting of the padded additional data $(A_1, \dots, A_a)$, the padded plaintext $(P_1, \dots, P_p)$ and a length block. The tag is then used as the synthetic IV for the CTR mode encryption of the plaintext.*

ticated encryption (Figure 2) takes as input three 128 bit keys K_1, K_2, K_3, a blocks of additional data $\boldsymbol{A} = (A_1, \dots, A_a)$, p blocks of plaintext data $\boldsymbol{P} = (P_1, \dots, P_p)$, a 128 bit nonce N and outputs the corresponding p ciphertext blocks $\boldsymbol{C} = (C_1, \dots, C_p)$ and a 128 bit message authentication tag T. The CCM-SIV authenticated decryption (Figure 3) takes as input again three 128 bit keys K_1, K_2, K_3, a blocks of additional data $\boldsymbol{A} = (A_1, \dots, A_a)$, p blocks of ciphertext $\boldsymbol{C} = (C_1, \dots, C_p)$, a 128 bit nonce N, a 128 bit message authentication tag T and outputs either the corresponding p blocks of plaintext $\boldsymbol{P} = (P_1, \dots, P_p)$ for a valid message, or the invalidity symbol $\perp$ for invalid messages. The three required keys K_1, K_2, K_3 for the underlying cryptographic primitives can be derived from a single master key K with a deterministic Pseudo Random Number Generator (PRNG) based on the AES block cipher [3, Section 4.4.4]. The key generation is shown in the first function of Algorithm 1.
The constant values $0, 1, 2$ (encoded as 128 bit unsigned integers) are encrypted using the master key K, to obtain the three required keys. In contrast to AES-GCM-SIV, these subkeys are fixed during the lifetime of a master key K. The key derivation has to be done only once after the negotiation of the master key K. If K is chosen uniformly at random, then E_K is a secure PRF. As long as the inputs to E_K are distinct, the corresponding outputs are uniformly distributed and independent. This is the case for constant values $0, 1, 2$.

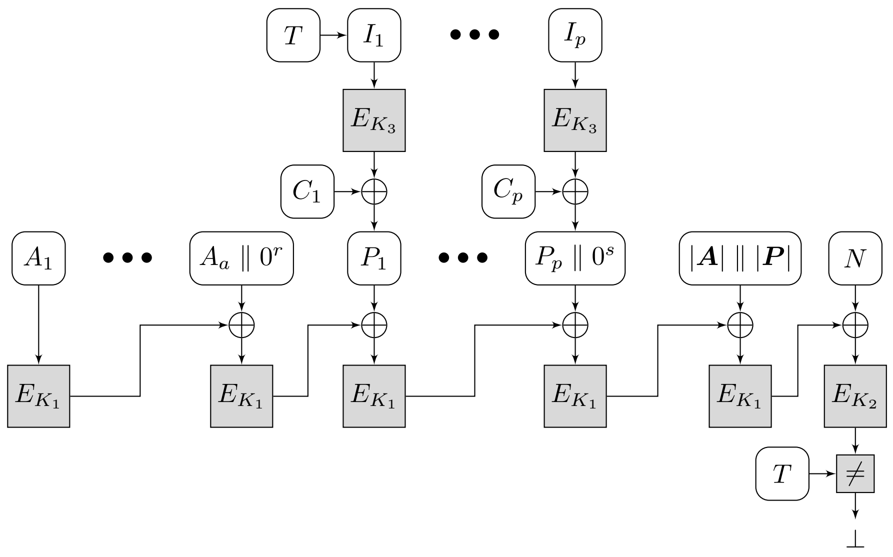


Fig. 3: *CCM-SIV decryption. First, the ciphertext $(C_1, \ldots, C_p)$ is decrypted by using the received tag T as the IV for CTR mode. The MAC is then calculated the same was as in the CCM-SIV encryption. The calculated tag is compared to the received tag T to check for validity.*

Algorithm 1 CCM-SIV

```
 1: function KeyGeneration( )
 2:     $K \xleftarrow{\$} \mathcal{K}$
 3:     $K_1 \leftarrow E_K(0)$
 4:     $K_2 \leftarrow E_K(1)$
 5:     $K_3 \leftarrow E_K(2)$
 6:     return $\boldsymbol{K} = (K_1, K_2, K_3)$

 7: function Encryption$_{\boldsymbol{K}}(\boldsymbol{A}, \boldsymbol{P}, N)$
 8:     $T \leftarrow E_{K_2}(\mathrm{CBC}_{K_1}(e(\boldsymbol{A}, \boldsymbol{P})) \oplus N)$
 9:     $\boldsymbol{C} \leftarrow \mathrm{CTR}^T_{K_3}(\boldsymbol{P})$
10:     return $(\boldsymbol{C}, T)$

11: function Decryption$_{\boldsymbol{K}}(\boldsymbol{A}, \boldsymbol{C}, N, T)$
12:     $\boldsymbol{P} \leftarrow \mathrm{CTR}^T_{K_3}(\boldsymbol{C})$
13:     $T' \leftarrow E_{K_2}(\mathrm{CBC}_{K_1}(e(\boldsymbol{A}, \boldsymbol{P})) \oplus N)$
14:     if $T = T'$ then
15:         return $\boldsymbol{P}$
16:     else
17:         return $\perp$
```

Authentication. The authentication is done by first applying an encoding function e on the input $(\boldsymbol{A}, \boldsymbol{P})$. This function appends r zeros to the last block A_a of the additional data and s zeros to the last block P_p of the plaintext. If any of these are already are multiple of the block size, then no zeros are appended. Similarly to AES-GCM-SIV, the encoding function adds another block called the length block. This block contains the concatenation of the original bit-length $|\boldsymbol{A}|$ with the original bit-length $|\boldsymbol{P}|$, both encoded as 64 bit unsigned integers yielding a full 128 bit block. To obtain the message tag T, the set of blocks $A_1, \ldots, (A_a \parallel 0^r), P_1, \ldots, (P_p \parallel 0^s), (|\boldsymbol{A}| \parallel |\boldsymbol{P}|)$ is applied to the CBC construction using key K_1. The output of the CBC construction is XORed with the nonce N and finally encrypted with a single block cipher invocation using key K_2.

Encryption/Decryption. The encryption/decryption is done in CTR mode by using the tag T as the IV. Here, the first counter block[1] I_1 is simply initialized with the tag T. Only the least significant 32 bit are used as the counter field.

This block (i.e. the IV) is simply initialized with the 128 bit tag T. For every subsequent counter block $I_2, I_3, \ldots$, the least significant 32 bits are incremented using $\bmod 2^{32}$ arithmetic. The maximum plaintext length for the CTR mode is therefore 2^{32} blocks, because then the 32 bit counter will wrap around, causing the two-time-pad problem. Note, that if the last plaintext block P_p is smaller than the blocksize, not the full keystream $E_K(I_p)$ is needed to encrypt/decrypt P_p resp. C_p. In the figures, the truncation of $E_K(I_p)$ to $|P_p|$ bits is omitted for reasons of simplicity. The latter two functions in Algorithm 1 describe the CCM-SIV authenticated encryption and authenticated decryption. These two functions are an instantiation of the general SIV paradigm shown in Figures 1(a) and 1(b) from the beginning of this section. Even though the CCM-SIV scheme is a constructed mixture of standard CCM, the SIV paradigm and the AES-GCM-SIV scheme - which are all proven secure - a proper security proof needs to be done for this mixture.

4 The Security of CCM-SIV

Based on the assumption, that the raw CBC-construction described by Algorithm 2 is a secure PRF for prefix-free messages [3, Section 6.4.1] and that the CTR mode-of-operation is a secure IV-based symmetric encryption scheme [7], the security proof of CCM-SIV is done in four steps:

1. The prefix-free PRF family CBC-E is an ϵ-almost XOR universal hash function family (ϵ-AXU) for messages, that are a multiple of the block size x. Since we cannot guarantee prefix-free messages, a less conservative model than a PRF is introduced here. It is shown, that the XOR-distance between two outputs of the CBC construction is almost truly random.
2. An injective encoding on the input yields an ϵ-AXU family for additional data and plaintext of any bit size.

[1]The first counter block is denoted as I_1, because there is no I_0 needed for the CTR encryption of the MAC, as it is the case for standard CCM.

Since messages are not always multiples of the block size, this step provides a solution to handle messages of any bit-length. Furthermore, vector-valued messages are introduced here.

3. The composition of this bit-wise ϵ-AXU, a XOR operation and a single block PRF yields a secure vector-input PRF over the additional data, the plaintext and the nonce.
   This step returns to the original notion of a PRF by composing several building blocks. Now, we have a PRF for vector inputs, that can be of any bit-size.
4. Combining the CTR mode-of-operation with the vector-input PRF from Step 3 yields a secure nonce-misuse-resistant authenticated encryption scheme.
   In this last step, the vector-input PRF from Step 3 and the CTR mode-of-operation are combined according to the SIV paradigm. It is proven, that this composition provides confidentiality, authenticity and integrity, even if the same nonce repeats.

4.1 Step 1: The raw CBC construction is an ϵ-AXU

In this step we prove, that the CBC-E construction is an ϵ-almost XOR universal hash function family for messages, that are a multiple of the block size x of the underlying PRF E.

Algorithm 2 CBC-E construction.

```
Require: $K \xleftarrow{\$} \mathcal{K}$

1: function CBC-$E_K(\boldsymbol{X})$          ▷ $\boldsymbol{X}$ consists of $m$ blocks.
2:     $Y \leftarrow 0$
3:     for $i \leftarrow 1$ to $m$ do
4:         $Y \leftarrow E_K(Y \oplus X_i)$
5:     return $Y$
```

The proof builds upon the fact, that the raw CBC construction CBC-E described by Algorithm 2 is a secure PRF family for prefix-free inputs, that are a multiple of the block size x [3, Section 6.4.1]. The set of all input messages is called *prefix-free*, if no message is a *proper prefix* of another message:

Definition 1. *[Proper Prefix] A message $\boldsymbol{X} = (X_1, \ldots, X_m)$ is a proper prefix of a message $\boldsymbol{X'} = (X'_1, \ldots, X'_{m'})$ with $1 \leq m < m' \leq m_{\max}$, if for every $1 \leq i \leq m$ it holds that $X_i = X'_i$.*

Following Boneh [3, Section 6.4.1], Definition 2 states that CBC-E_K is indistinguishable from a truly random function, if the messages are prefix-free and K is chosen uniformly from the key space $\mathcal{K}$. The related security game PF-PRF works as follows: The PPT adversary $\mathcal{A}$ queries her oracle $\mathcal{O}$ with at most q prefix-free messages $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_q \in \mathcal{X}^{\leq m_{\max}}$, where the oracle is either CBC-E_K with $K \xleftarrow{\$} \mathcal{K}$ or a truly random function with the same output size x. $\mathcal{A}$ wins the game, if she is able to distinguish both oracle versions with non-negligible probability. The wrapping PPT algorithm $\mathcal{B}$ plays the PRF game attacking the underlying PRF E_K, where he queries at most $m_{\max}$ message blocks for every $\mathcal{A}$-query. $\mathcal{B}$'s oracle is either the PRF E_K with a freshly chosen $K \xleftarrow{\$} \mathcal{K}$ or a truly random function with the same output size x. $\mathcal{B}$ wins the PRF game, if he is able to distinguish both oracle versions with non-negligible probability:

Definition 2. *[*CBC-*E's security as a PRF] Let $\mathcal{A}$ be a PPT algorithm, who is able to distinguish* CBC-*E from a random function in the* PF-PRF *game with non-negligible advantage. Then there exists a PPT adversary $\mathcal{B}$, who distinguishes E from a random function in the* PRF *game with non-negligible advantage. Particularly, it holds that*

$$\mathbf{adv}^{\text{PF-PRF}}_{\text{CBC-}E}(\mathcal{A}) \leq \mathbf{adv}^{\text{PRF}}_{E}(\mathcal{B}) + \frac{q^2 m_{\max}^2}{2 \cdot 2^x}. \tag{1}$$

Based on the assumption, that the AES block cipher E is secure, such an $\mathcal{A}$ cannot exist, since this would contradict (1).

Given a secure PRF family for prefix-free inputs that are a multiple of the block size x such as CBC-E, we now show that this has a property called *difference unpredictability*, *uniform difference property* or simply *ϵ-Almost XOR Universality.*

Definition 3. *[ϵ-Almost XOR Universal Hash Function Family.] An ϵ-Almost XOR Universal Hash Function Family (ϵ-AXU) is a function family $H : \mathcal{K} \times \{0,1\}^* \to \mathcal{X}$ with $H = \{H_K : \{0,1\}^* \to \mathcal{X} \mid \forall K \in \mathcal{K}\}$, $\mathcal{X} = \{0,1\}^x$, $\mathcal{K} = \{0,1\}^k$, such that for all $\boldsymbol{X}, \boldsymbol{X'} \in \{0,1\}^*, \boldsymbol{X} \neq \boldsymbol{X'}$, all H_K and all $\delta \in \mathcal{X}$ it holds that*

$$\Pr\left[H_K(\boldsymbol{X}) \oplus H_K(\boldsymbol{X'}) = \delta\right] \leq \epsilon$$

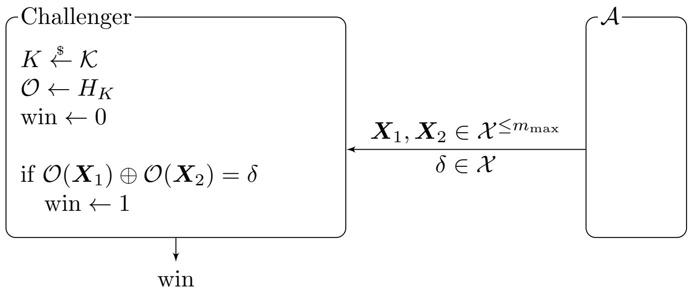
with $0 < \epsilon < 1$. Or in other words, if the probability, that the XOR-difference of two hash values is δ, is at most ϵ.

The related XUHF game defined in Figure 4 works as follows: In the beginning, the challenger chooses a key K uniformly from the key space $\mathcal{K}$ and then sets the oracle to H_K. The adversary sends two distinct messages $\boldsymbol{X}_1, \boldsymbol{X}_2 \in \mathcal{X}^{\leq m_{\max}}$ together with a guess $\delta \in \mathcal{X}$ for the XOR-difference of the hash values. The challenger checks, if $H_K(\boldsymbol{X}_1) \oplus H_K(\boldsymbol{X}_2) = \delta$ and sets the flag win in this case. Obviously, $\mathcal{A}$ wins the game if win $= 1$ and loses the game if win $= 0$. $\mathcal{A}$'s advantage in this game is defined to be $\Pr[\text{win} = 1]$ or short $\Pr[\text{win}]$. Upper-bounding this advantage then yields a measure for the security as an ϵ-AXU:

Definition 4. *[ϵ-AXU security.] Let H be a function family as in Definition 3. Then H is a computationally secure ϵ-Almost XOR UHF family (ϵ-AXU), if there exists a negligible ϵ, such that for all PPT adversaries $\mathcal{A}$ playing the* XUHF *game defined in Figure 4 it holds that*

$$\mathbf{adv}^{\text{XUHF}}_{H}(\mathcal{A}) \leq \epsilon.$$

The following Lemma proves, that CBC-E is also an ϵ-AXU family:



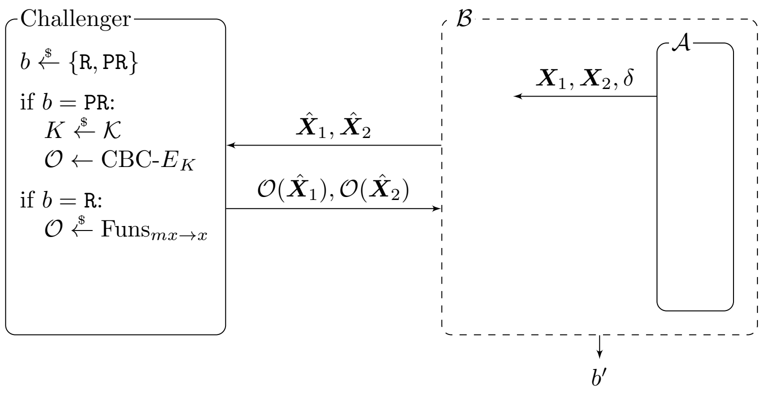
Challenger
$K \xleftarrow{\$} \mathcal{K}$
$\mathcal{O} \leftarrow H_K$
win $\leftarrow 0$
if $\mathcal{O}(\boldsymbol{X}_1) \oplus \mathcal{O}(\boldsymbol{X}_2) = \delta$
win $\leftarrow 1$
$\boldsymbol{X}_1, \boldsymbol{X}_2 \in \mathcal{X}^{\leq m_{\max}}$
$\delta \in \mathcal{X}$
$\mathcal{A}$
win

Fig. 4: *The challenger chooses a function family instance H_K by uniformly choosing a key K from the set $\mathcal{K}$. Then the adversary $\mathcal{A}$ outputs two distinct messages $\boldsymbol{X}_1, \boldsymbol{X}_2$ and a guess δ for the XOR-difference. The challenger evaluates H_K on $\boldsymbol{X}_1, \boldsymbol{X}_2$ and sets the flag* win *to 1, if δ was right.*

Lemma 1. *[*CBC-*E's security as an ϵ-AXU.] Let $\mathcal{A}$ be a PPT adversary, who wins the* XUHF *game defined in Figure 4 with non-negligible advantage. Then there exists a wrapping PPT adversary $\mathcal{B}$, who distinguishes* CBC-*E from a random function in the* PF-PRF *game with non-negligible advantage. Particularly, it holds that*

$$\mathbf{adv}^{\mathrm{XUHF}}_{\mathrm{CBC}\text{-}E}(\mathcal{A}) \leq \mathbf{adv}^{\mathrm{PF\text{-}PRF}}_{\mathrm{CBC}\text{-}E}(\mathcal{B}) + \frac{1}{2^x}.$$

Proof. The proof works by describing a wrapping adversary $\mathcal{B}$, which tries to win the PF-PRF game by making use of a hypothetical adversary $\mathcal{A}$, which is playing the XUHF game. Figure 5 shows the framework for $\mathcal{B}$. Adversary $\mathcal{A}$



Challenger
$b \xleftarrow{\$} \{\mathtt{R}, \mathtt{PR}\}$
if $b = \mathtt{PR}$:
$K \xleftarrow{\$} \mathcal{K}$
$\mathcal{O} \leftarrow$ CBC-E_K
if $b = \mathtt{R}$:
$\mathcal{O} \xleftarrow{\$} \mathrm{Funs}_{mx \to x}$
$\hat{\boldsymbol{X}}_1, \hat{\boldsymbol{X}}_2$
$\mathcal{O}(\hat{\boldsymbol{X}}_1), \mathcal{O}(\hat{\boldsymbol{X}}_2)$
$\mathcal{B}$
$\boldsymbol{X}_1, \boldsymbol{X}_2, \delta$
$\mathcal{A}$
b'

Fig. 5: *The wrapping algorithm $\mathcal{B}$ tries to win the* PF-PRF *game on the outside by using the* XUHF*-adversary $\mathcal{A}$ as a subroutine. $\mathcal{B}$ sets the flag guess b' to either* PR *or* R*, depending on whether $\mathcal{O}(\hat{\boldsymbol{X}}_1)$ and $\mathcal{O}(\hat{\boldsymbol{X}}_2)$ collide or not.*

is playing the XUHF game and therefore outputs two distinct messages $\boldsymbol{X}_1, \boldsymbol{X}_2$ together with an expected XOR-difference δ. Adversary $\mathcal{B}$'s goal is to distinguish between CBC-E and a random choice out of $\text{Funs}_{mx \to x}$. For this, $\mathcal{B}$ may do the following: First, he appends δ to $\boldsymbol{X}_1$ and a zero block to $\boldsymbol{X}_2$. With this, the modified messages $\hat{\boldsymbol{X}}_1 = \boldsymbol{X}_1 \| \delta$ and $\hat{\boldsymbol{X}}_2 = \boldsymbol{X}_2 \| 0^x$ will cause a detectable collision in $\mathcal{O}(\hat{\boldsymbol{X}}_1)$ and $\mathcal{O}(\hat{\boldsymbol{X}}_2)$, if $\mathcal{O} = \text{CBC-}E_K$ and δ was correctly guessed by $\mathcal{A}$. This is because $\delta = \text{CBC-}E_K(\boldsymbol{X}_1) \oplus \text{CBC-}E_K(\boldsymbol{X}_2)$ and therefore

$$\begin{aligned} &\text{CBC-}E_K(\hat{\boldsymbol{X}}_1) \quad \oplus \quad \text{CBC-}E_K(\hat{\boldsymbol{X}}_2) \\ = \quad & E_K(\text{CBC-}E_K(\boldsymbol{X}_1) \oplus \delta) \quad \oplus \quad E_K(\text{CBC-}E_K(\boldsymbol{X}_2) \oplus 0^x) \\ = \quad & E_K(\text{CBC-}E_K(\boldsymbol{X}_2)) \quad \oplus \quad E_K(\text{CBC-}E_K(\boldsymbol{X}_2) \oplus 0^x) \\ = \quad & 0 \end{aligned}$$

The problem with this is, that $\mathcal{B}$ cannot simply pass $\boldsymbol{X}_1 \| \delta$ and $\boldsymbol{X}_2 \| 0^x$ to his challenger, since this message pair might not be prefix-free. So $\mathcal{B}$ has to make the pair prefix-free first. For this, we can make use of the *extension property* of CBC-E [3, Section 7.2.2]:

Definition 5. *[Extension Property.] A PRF Family $F : \mathcal{K} \times \mathcal{X}^{\leq m_{\max}} \to \mathcal{X}$ with $F = \{F_K : \mathcal{X}^{\leq m_{\max}} \to \mathcal{X} \mid \forall K \in \mathcal{K}\}$, $\mathcal{X} = \{0,1\}^x$ and $K = \{0,1\}^k$ is "extendable" or has an "extension property", if*

$$F_K(\boldsymbol{X}) = F_K(\boldsymbol{X'}) \Rightarrow F_K(\boldsymbol{X} \| A) = F_K(\boldsymbol{X'} \| A)$$

for all $\boldsymbol{X}, \boldsymbol{X'} \in \mathcal{X}^{\leq m_{\max}}$, $\boldsymbol{X} \neq \boldsymbol{X'}$, all $A \in \mathcal{X}$ and all $K \in \mathcal{K}$.

Obviously, CBC-E is extendable: Consider two distinct messages $\boldsymbol{X}$ and $\boldsymbol{X}'$ with $\text{CBC-}E_K(\boldsymbol{X}) = \text{CBC-}E_K(\boldsymbol{X}')$ and some $A \in \mathcal{X}$. Then

$$\begin{aligned} \text{CBC-}E_K(\boldsymbol{X} \| A) &= \text{CBC-}E_K(\text{CBC-}E_K(\boldsymbol{X}) \oplus A) \\ &= \text{CBC-}E_K(\text{CBC-}E_K(\boldsymbol{X}') \oplus A) \\ &= \text{CBC-}E_K(\boldsymbol{X}' \| A). \end{aligned}$$

To make the two queries $\boldsymbol{X}_1 \| \delta$ and $\boldsymbol{X}_2 \| 0^x$ prefix-free, adversary $\mathcal{B}$ uses the extension property in the following way (the pseudo code for the wrapping adversary $\mathcal{B}$ is given in Algorithm 3).

Algorithm 3 Pseudo Code for $\mathcal{B}$.

```
1: receive $\boldsymbol{X}_1, \boldsymbol{X}_2, \delta$ from $\mathcal{A}$
2: choose any $A$ s.t. $A \notin \boldsymbol{X}_1, \boldsymbol{X}_2$ and $A \neq \delta$
3: $\hat{\boldsymbol{X}}_1 \leftarrow \boldsymbol{X}_1 \| \delta \| A$
4: $\hat{\boldsymbol{X}}_2 \leftarrow \boldsymbol{X}_2 \| 0^x \| A$
5: send $\hat{\boldsymbol{X}}_1, \hat{\boldsymbol{X}}_2$ to challenger
6: if $\mathcal{O}(\hat{\boldsymbol{X}}_1) = \mathcal{O}(\hat{\boldsymbol{X}}_2)$ then
7:     return $b' \leftarrow$ PR
8: else
9:     return $b' \leftarrow$ R
```

Upon receiving queries $\boldsymbol{X}_1, \boldsymbol{X}_2$ from $\mathcal{A}$ and appending δ and 0^x, $\mathcal{B}$ chooses a block A that is not an element of $\boldsymbol{X}_1$ or $\boldsymbol{X}_2$ and is neither δ nor 0^x. Such an A must exist in an asymptotic setting, because the PPT adversary $\mathcal{A}$ outputs at most a polynomially bounded number of blocks and the number of possible block values grows exponentially in x. This A is appended to both messages, yielding a prefix-free pair $\hat{\boldsymbol{X}}_1 = \boldsymbol{X}_1 \| \delta \| A$ and $\hat{\boldsymbol{X}}_2 = \boldsymbol{X}_2 \| 0^x \| A$. Adversary $\mathcal{B}$ then queries his challenger with this prefix-free pair. If the two messages $\boldsymbol{X}_1 \| \delta$ and $\boldsymbol{X}_2 \| 0^x$ would have caused a collision CBC-$E_K(\boldsymbol{X}_1 \| \delta) =$ CBC-$E_K(\boldsymbol{X}_2 \| 0^x)$, then due to the extension property of CBC-E, the extended messages $\hat{\boldsymbol{X}}_1 = \boldsymbol{X}_1 \| \delta \| A$ and $\hat{\boldsymbol{X}}_2 = \boldsymbol{X}_2 \| 0^x \| A$ must also cause a collision CBC-$E_K(\hat{\boldsymbol{X}}_1) =$ CBC-$E_K(\hat{\boldsymbol{X}}_2)$. Once he has sent the modified messages $\hat{\boldsymbol{X}}_1, \hat{\boldsymbol{X}}_2$ to his challenger and received $\mathcal{O}(\hat{\boldsymbol{X}}_1), \mathcal{O}(\hat{\boldsymbol{X}}_2)$, $\mathcal{B}$ checks if $\mathcal{O}(\hat{\boldsymbol{X}}_1)$ collides with $\mathcal{O}(\hat{\boldsymbol{X}}_2)$. If this is the case, he outputs `PR`. Otherwise, he outputs `R`. Since we are assuming that $\mathcal{A}$ guesses the right δ for CBC-$E_K(\boldsymbol{X}_1) \oplus$ CBC-$E_K(\boldsymbol{X}_2)$ with non-negligible probability, the constructed adversary $\mathcal{B}$ is then also able to distinguish CBC-E from a random function with non-negligible probability because of the following collision:

$$\begin{aligned} & E_K(E_K(\text{CBC-}E_K(\boldsymbol{X_1}) \oplus \delta) \oplus A) \\ = \;& E_K(E_K(\text{CBC-}E_K(\boldsymbol{X_1}) \oplus \text{CBC-}E_K(\boldsymbol{X_1}) \oplus \text{CBC-}E_K(\boldsymbol{X_2})) \oplus A) \\ = \;& E_K(E_K(\text{CBC-}E_K(\boldsymbol{X_2}) \oplus 0) \oplus A \end{aligned} \tag{2}$$

To relate the advantages of $\mathcal{A}$ and $\mathcal{B}$, first observe that

$$\begin{aligned} \mathbf{adv}^{\text{PF-PRF}}_{\text{CBC-}E}(\mathcal{B}) &= \left| \Pr\left[\mathcal{B}^{\text{Funs}_{mx\to x}} \Rightarrow \mathtt{R}\right] - \Pr\left[\mathcal{B}^{\text{CBC-}E} \Rightarrow \mathtt{R}\right] \right| \\ &= \left| 1 - \Pr\left[\mathcal{B}^{\text{Funs}_{mx\to x}} \Rightarrow \mathtt{PR}\right] - \left(1 - \Pr\left[\mathcal{B}^{\text{CBC-}E} \Rightarrow \mathtt{PR}\right]\right) \right| \\ &\geq \Pr\left[\mathcal{B}^{\text{CBC-}E} \Rightarrow \mathtt{PR}\right] - \Pr\left[\mathcal{B}^{\text{Funs}_{mx\to x}} \Rightarrow \mathtt{PR}\right]. \end{aligned} \tag{3}$$

If $\mathcal{B}$ is interacting with a uniform choice out of $\text{Funs}_{mx\to x}$, the values $\mathcal{O}(\hat{\boldsymbol{X}}_1)$ and $\mathcal{O}(\hat{\boldsymbol{X}}_2)$ are uniformly distributed in $\mathcal{X}$ and therefore

$$\mathcal{O}(\hat{\boldsymbol{X}}_1) = \mathcal{O}(\hat{\boldsymbol{X}}_2)$$

holds with probability $1/|\mathcal{X}|$ and thus

$$\Pr\left[\mathcal{B}^{\text{Funs}_{mx\to x}} \Rightarrow \mathtt{PR}\right] = \frac{1}{2^x}. \tag{4}$$

$\mathcal{A}$'s guessed difference δ is correct with probability $\mathbf{adv}^{\text{XUHF}}_{H}(\mathcal{A})$. If $\mathcal{B}$ is interacting with CBC-E, a correct δ causes the collision in (2). Hence $\mathcal{B}$ outputs `PR` with at least this probability:

$$\Pr\left[\mathcal{B}^{\text{CBC-}E} \Rightarrow \mathtt{PR}\right] \geq \mathbf{adv}^{\text{XUHF}}_{H}(\mathcal{A}). \tag{5}$$

Combining (3), (4) and (5) concludes the proof:

$$\mathbf{adv}^{\text{PF-PRF}}_{\text{CBC-}E}(\mathcal{B}) \geq \mathbf{adv}^{\text{XUHF}}_{H}(\mathcal{A}) - \frac{1}{2^x}.$$

$\square$

4.2 Step 2: Injective encoding for bit-wise vector-valued inputs

The function family CBC-$E : \mathcal{K} \times \mathcal{X}^{\leq m_{\max}} \to \mathcal{X}$ with $\mathcal{X} = \{0,1\}^x$ was proved to be a secure ϵ-AXU family for messages, that are a multiple of the block size x and at most $m_{\max}$ blocks long. This step constructs a secure ϵ-AXU family CBC-$E^* : \mathcal{K} \times \mathcal{A} \times \mathcal{P} \to \mathcal{X}$, that takes as input an additional data part $\boldsymbol{A} \in \mathcal{A}$ and a plaintext $\boldsymbol{P} \in \mathcal{P}$, where $\mathcal{A} = \mathcal{P} = \{0,1\}^*$, but with the following restriction on the maximum length of $\boldsymbol{A}$ and $\boldsymbol{P}$:

$$\left\lceil \frac{|\boldsymbol{A}|}{x} \right\rceil + \left\lceil \frac{|\boldsymbol{P}|}{x} \right\rceil + 1 \leq m_{\max}. \tag{6}$$

Without loss of generality, additionally assume that $|\boldsymbol{A}|, |\boldsymbol{P}| \leq 2^{x/2}$ bit, so that both the bit-length of $\boldsymbol{A}$ and the bit-length of $\boldsymbol{P}$ can be expressed as $x/2$ bit integer values.

Algorithm 4 Injective Encoding Function e.

```
1: function $e(\boldsymbol{A}, \boldsymbol{P})$
2:     $r \leftarrow x - |A_a|$
3:     $s \leftarrow x - |P_p|$
4:     $L \leftarrow |\boldsymbol{A}| \parallel |\boldsymbol{P}| : L \in \{0,1\}^x$
5:     $\boldsymbol{X} \leftarrow \boldsymbol{A} \parallel 0^r \parallel \boldsymbol{P} \parallel 0^s \parallel L$
6:     return $\boldsymbol{X} = (X_1, \ldots, X_m)$
```

Fix an input tuple $(\boldsymbol{A}, \boldsymbol{P})$ according to the restrictions described above and let $K \xleftarrow{\$} \mathcal{K}$, then CBC-$E^*_K(\boldsymbol{A}, \boldsymbol{P})$ is constructed as follows: CBC-$E^*_K(\boldsymbol{A}, \boldsymbol{P}) =$ CBC-$E_K(e(\boldsymbol{A}, \boldsymbol{P}))$, with an injective encoding function $e : \{0,1\}^* \times \{0,1\}^* \to \mathcal{X}^{\leq m_{\max}}$. The encoding function e is defined by Algorithm 4 and CBC-E^* is illustrated in Figure 6. The additional data $\boldsymbol{A}$ consists of a blocks $A_1, \ldots, A_a$,

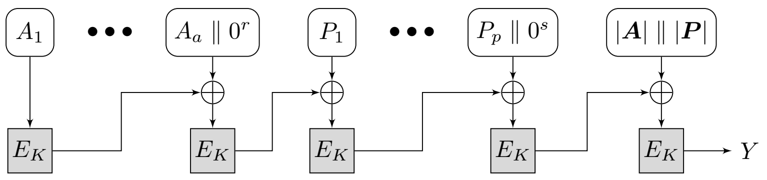


Fig. 6: *The* CBC-E^* *function is the composition of the injective encoding function e and the standard CBC construction. The additional data and the plaintext are padded with zeros and a length block containing the bit lengths of* $\boldsymbol{A}$ *and* $\boldsymbol{P}$ *is appended to the data, before applying the CBC construction.*

where the last block A_a might be incomplete, i.e. $|A_a| \leq x$. The same holds for $\boldsymbol{P} = (P_1, \ldots, P_p)$ with $|P_p| \leq x$. The injective encoding function e fills A_a with r zeros, such that $|(A_a \parallel 0^r)| = x$ and fills P_p with s zeros, such that $|(P_p \parallel 0^s)| = x$. These two parts are concatenated together with a length block

L that contains the bit-length of $\boldsymbol{A}$ and the bit-length of $\boldsymbol{P}$, both encoded as $x/2$ bit values. The final output $\boldsymbol{X}$ then contains m blocks with

$$m = a + p + 1 \leq m_{\max}$$

and hence is a proper input for CBC-E. The following corollary states, that CBC-E^* is also a secure ϵ-AXU family.

Corollary 1. *[*CBC-E^**'s security as an ϵ-AXU family.] If the* CBC-E *function family shown in Algorithm 2 is a secure ϵ-AXU family for messages that are a multiple of the block size x and that are at most $m_{\max}$ blocks long, then the composition* CBC-$E^*(\boldsymbol{A}, \boldsymbol{P}) =$ CBC-$E(e(\boldsymbol{A}, \boldsymbol{P}))$ *is also a secure ϵ-AXU family for distinct messages $(\boldsymbol{A}, \boldsymbol{P})$ that fulfills (6).*

Proof. First, let us verify that e is indeed an injective function, i.e. it never maps two distinct messages $(\boldsymbol{A}, \boldsymbol{P}), (\boldsymbol{A}', \boldsymbol{P}')$ to the same output $\boldsymbol{X} = \boldsymbol{X}'$. Consider two distinct messages $(\boldsymbol{A}, \boldsymbol{P}), (\boldsymbol{A}', \boldsymbol{P}')$ and their corresponding output $\boldsymbol{X}$ and $\boldsymbol{X}'$. We assume that $\boldsymbol{X} = \boldsymbol{X}'$. Then it must hold that $L = L'$, since these are the last blocks in $\boldsymbol{X}$ resp. $\boldsymbol{X}'$. Hence $|\boldsymbol{A}| = |\boldsymbol{A}'|$ and $|\boldsymbol{P}| = |\boldsymbol{P}'|$. From this follows, that $r = r'$ and $s = s'$. The only parts of $\boldsymbol{X}$ and $\boldsymbol{X}'$, that are left, are $\boldsymbol{A}$ and $\boldsymbol{P}$ itself. Therefore, they must be equal if $\boldsymbol{X} = \boldsymbol{X}'$. This contradicts the assumption, that $(\boldsymbol{A}, \boldsymbol{P}), (\boldsymbol{A}', \boldsymbol{P}')$ are distinct. Therefore, e is an injective function, which preserves the distinctness between two messages. Furthermore, e never outputs messages that are longer than $m_{\max}$ blocks due to its restriction on $\boldsymbol{A}$ and $\boldsymbol{P}$. By Lemma 1, applying the CBC-E construction on distinct inputs with a maximum length of $m_{\max}$ blocks yields a secure ϵ-AXU family. $\square$

4.3 Step 3: Construction of a PRF for MAC calculation

For the construction of a PRF over a nonce $N \in \mathcal{X}$, additional data $\boldsymbol{A} \in \{0,1\}^*$ and plaintext $\boldsymbol{P} \in \{0,1\}^*$, this step follows the same approach as Gueron and Lindell used for GCM-SIV [7]. There, the PRF is constructed by the composition $F_{K_2}(H_{K_1}(e(\boldsymbol{A}, \boldsymbol{P})) \oplus N)$, where $F_{K_2} : \mathcal{X} \to \mathcal{X}$ is a PRF instance using an encryption key $K_2 \xleftarrow{\$} \mathcal{K}$, $H_{K_1} : \mathcal{X}^{\leq m_{\max}} \to \mathcal{X}$ is an ϵ-AXU instance using a hashing key $K_1 \xleftarrow{\$} \mathcal{K}$ and $e : \{0,1\}^* \times \{0,1\}^* \to \mathcal{X}^{\leq m_{\max}}$ is the injective encoding function from the previous step. For CCM-SIV, the ϵ-AXU family H is CBC-E^* and the PRF F is the AES block cipher E. The composition $E_{K_2}($CBC-$E_{K_1}(e(\boldsymbol{A}, \boldsymbol{P})) \oplus N)$ is denoted as $F^*_{K_1,K_2}$ and illustrated in Figure 7. Formally, $F^* : \mathcal{K}^2 \times \mathcal{A} \times \mathcal{P} \times \mathcal{N} \to \mathcal{T}$ with $\mathcal{A} = \mathcal{P} = \{0,1\}^*$, $\mathcal{N} = \mathcal{T} = \mathcal{K} = \mathcal{X}$ and $F^* = \{F^*_{K_1,K_2} : \mathcal{A} \times \mathcal{P} \times \mathcal{N} \to \mathcal{T} \mid \forall K_1, K_2 \in \mathcal{K}\}$. The message $\boldsymbol{X} = (X_1, \ldots, X_m)$ in Figure 7 is the output of the injective encoding function e and contains the additional data $\boldsymbol{A}$, the plaintext $\boldsymbol{P}$ and the length block L. For a blocks of additional data $\boldsymbol{A}$ and p blocks of plaintext $\boldsymbol{P}$, it holds that $m = a + p + 1 \leq m_{\max}$. The output of CBC-$E_{K_1}$ is XORed with the nonce N and then encrypted by E_{K_2} to obtain the tag T. To analyze the security of this composition as a vector-input PRF for bit-wise inputs, Figure 8 defines the

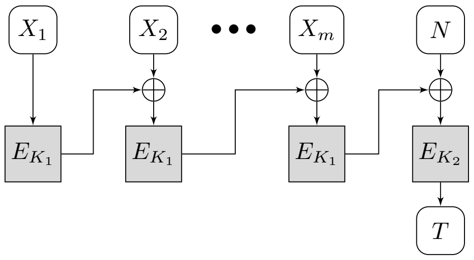


Fig. 7: *The secure PRF family F^* over the nonce N and a message $(X_1, \ldots, X_m)$ is accomplished by a composition of* CBC-E_{K_1} *as an ϵ-AXU, a XOR operation and a PRF E_{K_2}.*

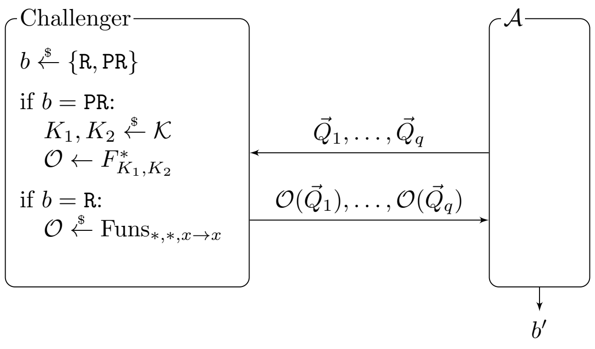

Challenger
$b \xleftarrow{\$} \{\mathtt{R}, \mathtt{PR}\}$
if $b = \mathtt{PR}$:
$K_1, K_2 \xleftarrow{\$} \mathcal{K}$
$\mathcal{O} \leftarrow F^*_{K_1,K_2}$
if $b = \mathtt{R}$:
$\mathcal{O} \xleftarrow{\$} \text{Funs}_{*,*,x\to x}$
$\mathcal{A}$
$\vec{Q}_1, \ldots, \vec{Q}_q$
$\mathcal{O}(\vec{Q}_1), \ldots, \mathcal{O}(\vec{Q}_q)$
b'

Fig. 8: *The oracle is either the PRF $F^*_{K_1,K_2}$ with K_1, K_2 chosen uniformly and independent or a uniform choice out of* $\text{Funs}_{*,*,x\to x}$*. The adversary $\mathcal{A}$ sends at most q vector valued queries $\vec{Q}_1, \ldots, \vec{Q}_q$ to the oracle and outputs a flag guess b' at the end.*

related security game PRF*. In this game, the adversary $\mathcal{A}$ outputs q distinct vector valued queries $\vec{Q}_1, \dots, \vec{Q}_q$, of which each $\vec{Q}_i$ with $1 \leq i \leq q$ is of the form $\vec{Q}_i = (\boldsymbol{A}_i, \boldsymbol{P}_i, N_i)$. The challenger sets the oracle to either a PRF instance $F^*_{K_1,K_2}$ with K_1, K_2 chosen uniformly and independently from the key space, or to a truly random function. The symbol $\mathrm{Funs}_{*,*,x\to x}$ denotes the set of all functions mapping $\{0,1\}^* \times \{0,1\}^* \times \mathcal{X} \to \mathcal{X}$. $\mathcal{A}$'s advantage is defined the same way as in the PRF game:

$$\mathbf{adv}_{F^*}^{\mathrm{PRF}^*}(\mathcal{A}) = \left|\Pr\left[\mathcal{A}^{\mathrm{Funs}_{*,*,x\to x}} \Rightarrow \mathtt{R}\right] - \Pr\left[\mathcal{A}^{F^*} \Rightarrow \mathtt{R}\right]\right|.$$

If this term can be upper-bounded by a negligible function for all PPT adversaries $\mathcal{A}$, then F^* is considered to be a secure vector-input PRF over the additional data $\boldsymbol{A}$, the plaintext $\boldsymbol{P}$ and the nonce N.

Lemma 2. *[F^*'s security as a PRF.] Let F^* be the composition of an ϵ-AXU family and a PRF family E as described above, and let $\mathcal{A}$ be a PPT adversary, who wins the* PRF* *game defined in Figure 8 with non-negligible advantage. Then there exists a wrapping PPT adversary $\mathcal{B}$, who distinguishes E from a random function in the* PRF *game. Particularly, it holds that*

$$\mathbf{adv}_{F^*}^{\mathrm{PRF}^*}(\mathcal{A}) \leq \mathbf{adv}_{E}^{\mathrm{PRF}}(\mathcal{B}) + \epsilon \cdot \frac{q^2}{2}.$$

Lemma 2 states, that an efficient adversary $\mathcal{A}$ breaking F^* cannot exist, because otherwise we would be able to construct an efficient adversary $\mathcal{B}$ breaking E and this would be a contradiction to the assumption, that AES is secure.

Proof (Proof of Lemma 2.). Figure 9 shows the framework for $\mathcal{B}$ and Algorithm 5 his pseudo code. The adversary $\mathcal{B}$ chooses the key K_1 for CBC-E by himself. When he receives the ith query $\vec{Q}_i$ from $\mathcal{A}$, he encodes $\boldsymbol{A}_i, \boldsymbol{P}_i$ using the injective encoding function e, applies CBC-E_{K_1}, XORs the resulting hash value with the nonce N_i and sends the result to his PRF challenger.

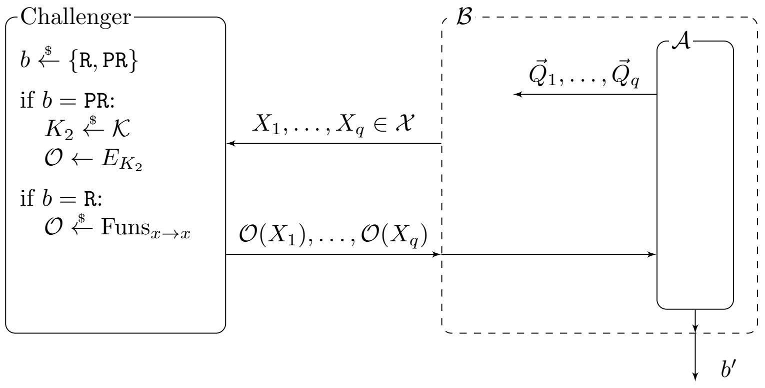
Algorithm 5 Pseudo Code for $\mathcal{B}$.

```
1: $K_1 \xleftarrow{\$} \mathcal{K}$
2: for all queries $\vec{Q}_i$ from $\mathcal{A}$ with $1 \leq i \leq q$ do
3:     $X_i \leftarrow$ CBC-$E_{K_1}(e(\boldsymbol{A}_i, \boldsymbol{P}_i)) \oplus N_i$
4:     send $X_i$ to challenger
5:     send $\mathcal{O}(X_i)$ to $\mathcal{A}$
6: return same $b'$ as $\mathcal{A}$
```

Depending on b, the challenger either applies E_{K_2} or a truly random function and sends back $\mathcal{O}(X_i)$. $\mathcal{B}$ then simply passes $\mathcal{O}(X_i)$ to $\mathcal{A}$. At the end, $\mathcal{B}$ outputs the same flag b' as $\mathcal{A}$. By definition, $\mathcal{B}$'s advantage in the PRF game and $\mathcal{A}$'s advantage in the PRF* game are

$$\mathbf{adv}_{E}^{\mathrm{PRF}}(\mathcal{B}) = \left|\Pr\left[\mathcal{B}^{\mathrm{Funs}_{x\to x}} \Rightarrow \mathtt{R}\right] - \Pr\left[\mathcal{B}^{E} \Rightarrow \mathtt{R}\right]\right|, \tag{7}$$

$$\mathbf{adv}_{F^*}^{\mathrm{PRF}^*}(\mathcal{A}) = \left|\Pr\left[\mathcal{A}^{\mathrm{Funs}_{*,*,x\to x}} \Rightarrow \mathtt{R}\right] - \Pr\left[\mathcal{A}^{F^*} \Rightarrow \mathtt{R}\right]\right|. \tag{8}$$



Challenger
$b \xleftarrow{\$} \{\mathtt{R}, \mathtt{PR}\}$
if $b = \mathtt{PR}$:
$K_2 \xleftarrow{\$} \mathcal{K}$
$\mathcal{O} \leftarrow E_{K_2}$
if $b = \mathtt{R}$:
$\mathcal{O} \xleftarrow{\$} \mathrm{Funs}_{x \to x}$
$\mathcal{B}$
$\mathcal{A}$
$\vec{Q}_1, \ldots, \vec{Q}_q$
$X_1, \ldots, X_q \in \mathcal{X}$
$\mathcal{O}(X_1), \ldots, \mathcal{O}(X_q)$
b'

Fig. 9: *The wrapping algorithm $\mathcal{B}$ tries to win the* PRF *game on the outside by using the* PRF**-adversary $\mathcal{A}$ as a subroutine. $\mathcal{B}$ outputs the same flag guess b' as $\mathcal{A}$.*

If $b = \mathtt{PR}$, $\mathcal{B}$ perfectly simulates the function $F^*_{K_1,K_2}$, because in this case

$$\mathcal{O}(X_i) = E_{K_2}(\text{CBC-}E_{K_1}(e(\boldsymbol{A}_i, \boldsymbol{P}_i)) \oplus N_i) \overset{\text{def}}{=} F^*_{K_1,K_2}$$

with K_1, K_2 chosen uniformly and independently from $\mathcal{K}$. Since $\mathcal{B}$ outputs the same flag b' as $\mathcal{A}$, it holds that

$$\Pr\left[\mathcal{B}^E \Rightarrow \mathtt{R}\right] = \Pr\left[\mathcal{A}^{F^*} \Rightarrow \mathtt{R}\right]. \tag{9}$$

Subtracting (8) and (7), applying the triangle inequality and using (9) yields

$$\begin{aligned}
&\mathbf{adv}^{\mathrm{PRF}^*}_{F^*}(\mathcal{A}) - \mathbf{adv}^{\mathrm{PRF}}_{E}(\mathcal{B}) \\
&\le \Bigg|\left|\Pr\left[\mathcal{A}^{\mathrm{Funs}_{*,*,x\to x}} \Rightarrow \mathtt{R}\right] - \Pr\left[\mathcal{A}^{F^*} \Rightarrow \mathtt{R}\right]\right| \\
&\quad - \left|\Pr\left[\mathcal{B}^{\mathrm{Funs}_{x\to x}} \Rightarrow \mathtt{R}\right] - \Pr\left[\mathcal{B}^{E} \Rightarrow \mathtt{R}\right]\right|\Bigg| \\
&\le \left|\Pr\left[\mathcal{A}^{\mathrm{Funs}_{*,*,x\to x}} \Rightarrow \mathtt{R}\right] - \Pr\left[\mathcal{B}^{\mathrm{Funs}_{x\to x}} \Rightarrow \mathtt{R}\right]\right|.
\end{aligned} \tag{10}$$

If $b = \mathtt{R}$, $\mathcal{B}$ perfectly simulates a uniform choice out of $\mathrm{Funs}_{*,*,x\to x}$, if and only if $\mathcal{O}(X_1), \ldots, \mathcal{O}(X_q)$ are all independent and uniformly distributed. This is only the case, if the inputs $X_1, \ldots, X_q$ for the function $\mathcal{O} \xleftarrow{\$} \mathrm{Funs}_{x\to x}$ are all distinct. Let coll denote the event, that there exists at least one pair of indices i, j with $i \neq j$ and $i, j \in \{1, \ldots, q\}$ such that

$$X_i = X_j \quad \Leftrightarrow \text{CBC-}E_{K_1}(e(\boldsymbol{A}_i, \boldsymbol{P}_i)) \oplus N_i = \text{CBC-}E_{K_1}(e(\boldsymbol{A}_j, \boldsymbol{P}_j)) \oplus N_j.$$

If coll does not occur, then all inputs to $\mathcal{O} \xleftarrow{\$} \mathrm{Funs}_{x\to x}$ are distinct and $\mathcal{B}$ perfectly simulates a uniform choice out of $\mathrm{Funs}_{*,*,x\to x}$. This is, because the output distribution of $\mathcal{O} \xleftarrow{\$} \mathrm{Funs}_{x \to x}$ is uniformly distributed. Using this conditional probability space, the absolute difference in (10) can be rewritten as

$$\begin{aligned}
&\left|\Pr\left[\mathcal{A}^{\mathrm{Funs}_{*,*,x\to x}} \Rightarrow \mathtt{R}\right] - \Pr\left[\mathcal{B}^{\mathrm{Funs}_{x\to x}} \Rightarrow \mathtt{R}\right]\right| \\
=&\left|\Pr\left[\mathcal{A}^{\mathrm{Funs}_{*,*,x\to x}} \Rightarrow \mathtt{R} \mid \mathrm{coll}\right]\Pr\left[\mathrm{coll}\right] + \Pr\left[\mathcal{A}^{\mathrm{Funs}_{*,*,x\to x}} \Rightarrow \mathtt{R} \mid \overline{\mathrm{coll}}\right]\Pr\left[\overline{\mathrm{coll}}\right]\right. \\
&\left. - \Pr\left[\mathcal{B}^{\mathrm{Funs}_{x\to x}} \Rightarrow \mathtt{R} \mid \mathrm{coll}\right]\Pr\left[\mathrm{coll}\right] - \Pr\left[\mathcal{B}^{\mathrm{Funs}_{x\to x}} \Rightarrow \mathtt{R} \mid \overline{\mathrm{coll}}\right]\Pr\left[\overline{\mathrm{coll}}\right]\right| \\
=&\Pr\left[\mathrm{coll}\right]\left|\Pr\left[\mathcal{A}^{\mathrm{Funs}_{*,*,x\to x}} \Rightarrow \mathtt{R} \mid \mathrm{coll}\right] - \Pr\left[\mathcal{B}^{\mathrm{Funs}_{x\to x}} \Rightarrow \mathtt{R} \mid \mathrm{coll}\right]\right| \\
\leq&\Pr\left[\mathrm{coll}\right].
\end{aligned} \tag{11}$$

The last inequality is because the absolute difference between two probabilities is at most 1. Combining (10) and (11) yields

$$\mathbf{adv}_{F^*}^{\mathrm{PRF}^*}(\mathcal{A}) - \mathbf{adv}_{E}^{\mathrm{PRF}}(\mathcal{B}) \leq \Pr\left[\mathrm{coll}\right]. \tag{12}$$

It remains to upper-bound $\Pr\left[\mathrm{coll}\right]$. For this, we are interested in the probability, that any two distinct $\mathcal{A}$ queries $\vec{Q}_i \neq \vec{Q}_j$ result in two same $\mathcal{B}$ queries $X_i = X_j$. We have to consider two cases:

1. $(\boldsymbol{A}_i, \boldsymbol{P}_i) = (\boldsymbol{A}_j, \boldsymbol{P}_j)$ and $N_i \neq N_j$: The probability that any two distinct queries collide can be written as

$$\begin{aligned}
&\Pr\left[\text{CBC-}E_{K_1}(e(\boldsymbol{A}_i, \boldsymbol{P}_i)) \oplus N_i = \text{CBC-}E_{K_1}(e(\boldsymbol{A}_j, \boldsymbol{P}_j)) \oplus N_j\right] \\
=&\Pr\left[\text{CBC-}E_{K_1}(e(\boldsymbol{A}_i, \boldsymbol{P}_i)) \oplus \text{CBC-}E_{K_1}(e(\boldsymbol{A}_j, \boldsymbol{P}_j)) = N_i \oplus N_j\right] \\
=&\ 0,
\end{aligned}$$

   because $N_i \oplus N_j \neq 0$ if $N_i \neq N_j$.
2. $(\boldsymbol{A}_i, \boldsymbol{P}_i) \neq (\boldsymbol{A}_j, \boldsymbol{P}_j)$ and $N_i = N_j$: Since CBC-E is an ϵ-AXU family, the probability that any two distinct queries collide can be written as

$$\begin{aligned}
&\Pr\left[\text{CBC-}E_{K_1}(e(\boldsymbol{A}_i, \boldsymbol{P}_i)) \oplus N_i = \text{CBC-}E_{K_1}(e(\boldsymbol{A}_j, \boldsymbol{P}_j)) \oplus N_j\right] \\
=&\Pr\left[\text{CBC-}E_{K_1}(e(\boldsymbol{A}_i, \boldsymbol{P}_i)) \oplus \text{CBC-}E_{K_1}(e(\boldsymbol{A}_j, \boldsymbol{P}_j)) = N_i \oplus N_j\right] \\
=&\Pr\left[\text{CBC-}E_{K_1}(e(\boldsymbol{A}_i, \boldsymbol{P}_i)) \oplus \text{CBC-}E_{K_1}(e(\boldsymbol{A}_j, \boldsymbol{P}_j)) = 0\right] \\
\leq&\ \epsilon,
\end{aligned}$$

   because the probability that the XOR-difference between two outputs of an ϵ-AXU family takes a specific value is at most ϵ.

Note, that there are less than $q^2/2$ possible pairs $\vec{Q}_i, \vec{Q}_j$. By the union bound it follows that

$$\Pr\left[\mathrm{coll}\right] \leq \epsilon \cdot \frac{q^2}{2}.$$

Applying this to (12) proves the lemma:

$$\mathbf{adv}_{F^*}^{\mathrm{PRF}^*}(\mathcal{A}) \leq \mathbf{adv}_{E}^{\mathrm{PRF}}(\mathcal{B}) + \epsilon \cdot \frac{q^2}{2}.$$

□

4.4 Step 4: Combining the PRF with an ivE according to SIV

The previous step established a secure vector-valued PRF F^* over the additional data, the plaintext and the nonce. This PRF is used to provide authenticity and integrity in CCM-SIV. In this step, we prove that combining this PRF with a secure IV-based symmetric encryption scheme for confidentiality yields a secure nonce-misuse-resistant authenticated encryption scheme.

Algorithm 6 CTR-E Encryption.

```
1: function CTR-$E^I_{K_3}(\boldsymbol{P})$
2:     $I_1 \leftarrow I$
3:     for $i \leftarrow 1$ to $p$ do
4:         $C_i \leftarrow E_{K_3}(I_i) \oplus P_i$
5:         $I_{i+1} \leftarrow \mathrm{inc}_{32}(I_i)$     ▷ Increments least significant
   32 bit of $I_i$ (modulo $2^{32}$).
6:     return $\boldsymbol{C} = (C_1, \ldots, C_p)$
```

CCM-SIV uses the CTR mode-of-operation of the block cipher E_K (cf. Algorithm 6) as the IV-based symmetric encryption scheme Π. Formally, Π is defined to be a set of three PPT algorithms (Gen, Enc, Dec) shown in Algorithm 7.

Algorithm 7 ivE scheme Π

```
1: function Gen( )
2:     return $K_3 \xleftarrow{\$} \mathcal{K}$

3: function Enc$_{K_3}(\boldsymbol{P})$
4:     $I \xleftarrow{\$} \mathcal{I}$
5:     $\boldsymbol{C} \leftarrow$ CTR-$E^I_{K_3}(\boldsymbol{P})$
6:     return $(\boldsymbol{C}, I)$

7: function Dec$_{K_3}(\boldsymbol{C}, I)$
8:     $\boldsymbol{P} \leftarrow$ CTR-$E^I_{K_3}(\boldsymbol{C})$
9:     return $\boldsymbol{P}$
```

The security game ivE for Π works as follows: The adversary $\mathcal{A}$ queries the oracle with at most q plaintext queries $\boldsymbol{P}_1, \ldots, \boldsymbol{P}_q \in \mathcal{X}^{\leq p_{\max}}$, where $p_{\max}$ is the maximum number of message blocks for a single message. For CCM-SIV $p_{\max} = 2^{32}$. The oracle is either the encryption function Enc_{K_3} with $K_3 \xleftarrow{\$} \mathcal{K}$ or a truly random function with the same output size. $\mathcal{A}$ wins the game, if she distinguishes both oracle versions with non-negligible probability. The security of Π can be reduced to the security of the underlying PRF E [7]:

Definition 6. *[$\mathit{\Pi}$'s security in the* ivE *game.] Let $\mathit{\Pi}$ be the ivE scheme defined by Algorithm 7 and $\mathcal{A}$ be an adversary, who wins the* ivE *game with non-negligible advantage. Then there exists a wrapping PPT adversary $\mathcal{B}$, who distinguishes E from a random function in the* PRF *game with non-negligible advantage. Particularly, it holds that*

$$\mathbf{adv}_{\mathit{\Pi}}^{\mathrm{ivE}}(\mathcal{A}) \leq \mathbf{adv}_{E}^{\mathrm{PRF}}(\mathcal{B}) + \frac{q^2 \cdot p_{\max}}{2 \cdot 2^x}.$$

Based on Definition 6 and Lemma 2, we construct a secure nonce-based authenticated encryption (nAE) scheme $\widetilde{\mathit{\Pi}}$ through a composition of $\mathit{\Pi}$ resp. CTR-E and F^* by using the 3-key instantiation of the SIV paradigm [18], also called construction A4 by Namprempre *et al.* [16]. The nAE scheme $\widetilde{\mathit{\Pi}}$ is a triple $(\widetilde{\mathrm{Gen}}, \widetilde{\mathrm{Enc}}, \widetilde{\mathrm{Dec}})$ of three PPT algorithms. The key generation algorithm $\widetilde{\mathrm{Gen}}$ generates a triple key $\boldsymbol{K} = (K_1, K_2, K_3)$, where each element K_1, K_2, K_3 is chosen uniformly from $\mathcal{K}$. The authenticated encryption algorithm $\widetilde{\mathrm{Enc}}$ takes as input the key $\boldsymbol{K}$, the additional data $\boldsymbol{A} \in \{0,1\}^*$, the plaintext $\boldsymbol{P} \in \{0,1\}^*$ and a nonce $N \in \mathcal{X}$ and outputs the corresponding ciphertext $\boldsymbol{C} \in \{0,1\}^*$ of the same length as $\boldsymbol{P}$ and a message tag $T \in \mathcal{X}$. The authenticated decryption algorithm $\widetilde{\mathrm{Dec}}$ takes as inputs the key $\boldsymbol{K}$, the additional data $\boldsymbol{A} \in \{0,1\}^*$, the ciphertext $\boldsymbol{C} \in \{0,1\}^*$, a nonce $N \in \mathcal{X}$ and a tag $T \in \mathcal{X}$. Without loss of generality, we assume that $\boldsymbol{P}$'s and $\boldsymbol{C}$'s sizes are a multiple of the block size x. Algorithm 8 gives the pseudo codes for $\widetilde{\mathit{\Pi}}$.

Algorithm 8 CCM-SIV scheme $\widetilde{\mathit{\Pi}}$ (formal description)

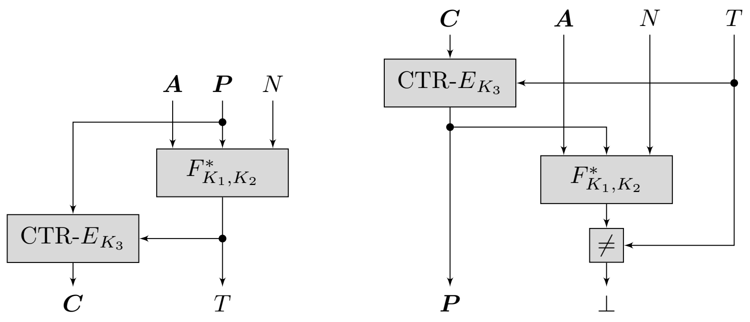
```
1: function $\widetilde{\mathrm{Gen}}()$
2:     return $\boldsymbol{K} \xleftarrow{\$} \mathcal{K}^3$

3: function $\widetilde{\mathrm{Enc}}_{\boldsymbol{K}}(\boldsymbol{A}, \boldsymbol{P}, N)$
4:     $T \leftarrow F^*_{K_1,K_2}(\boldsymbol{A}, \boldsymbol{P}, N)$
5:     $\boldsymbol{C} \leftarrow \text{CTR-}E^T_{K_3}(\boldsymbol{P})$
6:     return $(\boldsymbol{C}, T)$

7: function $\widetilde{\mathrm{Dec}}_{\boldsymbol{K}}(\boldsymbol{A}, \boldsymbol{C}, N, T)$
8:     $\boldsymbol{P} \leftarrow \text{CTR-}E^T_{K_3}(\boldsymbol{C})$
9:     $T' \leftarrow F^*_{K_1,K_2}(\boldsymbol{A}, \boldsymbol{P}, N)$
10:    if $T = T'$ then
11:        return $\boldsymbol{P}$
12:    else
13:        return $\perp$
```

Figure 10(a) and 10(b) provide a graphical representation of $\widetilde{\mathrm{Enc}}$ and $\widetilde{\mathrm{Dec}}$.

To analyze the security of $\widetilde{\mathit{\Pi}}$, we let $\mathcal{A}$ play the nAE security game defined in Figure 11. In the beginning, the challenger again chooses a flag b uniformly from the set $\{\mathtt{R}, \mathtt{PR}\}$. But this time, the challenger establishes two types of oracles, the authenticated encryption oracle $\mathcal{O}_e$ and the authenticated decryption oracle $\mathcal{O}_d$. In the pseudo-random case, the challenger runs the key generation $\widetilde{\mathrm{Gen}}$ to



(a) CCM-SIV $\widetilde{\text{Enc}}$ (b) CCM-SIV $\widetilde{\text{Dec}}$

Fig. 10: *For the encryption (a), the tag T is calculated over the additional data, the plaintext and the nonce by applying the vector-input PRF $F^*_{K_1,K_2}$. The tag T is then used as the IV for the CTR encryption function* CTR-E_{K_3} *to encrypt the plaintext. For the decryption (b), the ciphertext is decrypted first by using the received tag T as the IV for the CTR mode decryption function* CTR-E_{K_3}. *Then the tag is calculated over the additional data, the corresponding plaintext and the nonce by evaluating the vector-input PRF $F^*_{K_1,K_2}$. If both tags are not equal, an invalidity symbol $\perp$ is set.*

generate the required keys $(K_1, K_2, K_3) = \boldsymbol{K}$. After that, he sets the encryption oracle to $\widetilde{\text{Enc}}$ and the decryption oracle to $\widetilde{\text{Dec}}$. In the random version of the game, the encryption oracle $\mathcal{O}_e$ and the decryption oracle $\mathcal{O}_d$ should represent the idealized versions of $\widetilde{\text{Enc}}$ resp. $\widetilde{\text{Dec}}$. For $\widetilde{\text{Enc}}$, the idealized version is a function, that takes as input the triple $(\boldsymbol{A}, \boldsymbol{P}, N)$ and outputs a truly random ciphertext-tag pair $(\boldsymbol{C}, T)$, where $\boldsymbol{C}$ is of the same length as $\boldsymbol{P}$. In other words, the encryption oracle $\mathcal{O}_e$ should be a random choice out of $\text{Funs}_{*,*,x\to*,x}$. To avoid ambiguities with the idealized version of the ivE scheme Π, the idealized version of $\widetilde{\text{Enc}}$ is denoted as the function $\$(\cdot)$, such that $\$ \xleftarrow{\$} \text{Funs}_{*,*,x\to*,x}$. The idealized version of $\widetilde{\text{Dec}}$ is the function $\perp(\cdot)$, which rejects every decryption query by outputting the invalidity symbol $\perp$. Throughout the game, the adversary sends two types of queries. She is allowed to send up to q_e encryption queries $\vec{Q}^e_1, \ldots, \vec{Q}^e_{q_e}$ to the encryption oracle $\mathcal{O}_e$, where each encryption query $\vec{Q}^e_i$ with $1 \leq i \leq q_e$ is of the form $\vec{Q}^e_i = (\boldsymbol{A}_i, \boldsymbol{P}_i, N_i)$ and she is allowed to send up to q_d decryption queries $\vec{Q}^d_1, \ldots, \vec{Q}^d_{q_d}$ to the decryption oracle $\mathcal{O}_d$, where each decryption query $\vec{Q}^d_j$ with $1 \leq j \leq q_d$ is of the form $\vec{Q}^d_j = (\boldsymbol{A}_j, \boldsymbol{C}_j, N_j, T_j)$. The encryption and decryption queries may be sent in any order by the adversary. At the end, $\mathcal{A}$ outputs a flag guess b'. It is assumed, that $\mathcal{A}$'s queries are all distinct. This is sound, because $\mathcal{A}$ learns nothing by querying the same message twice, because she receives the same output twice. Note, that repeating nonces are explicitly allowed in the encryption queries, as long as the triples $(\boldsymbol{A}_i, \boldsymbol{P}_i, N_i)$ are all distinct. This is, because we want to construct a nonce-misuse-resistant

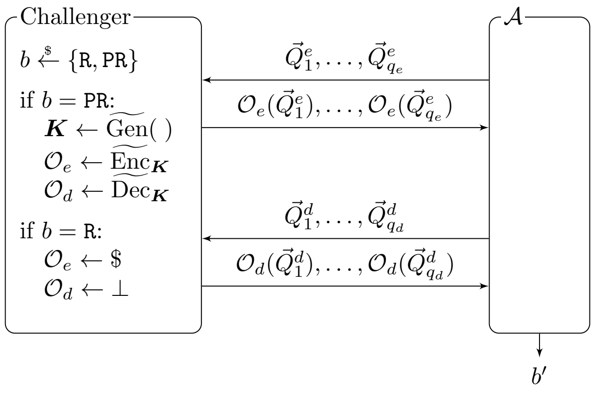

Challenger
$b \xleftarrow{\$} \{\mathtt{R}, \mathtt{PR}\}$
if $b = \mathtt{PR}$:
$\boldsymbol{K} \leftarrow \widetilde{\mathrm{Gen}}(\,)$
$\mathcal{O}_e \leftarrow \widetilde{\mathrm{Enc}}_{\boldsymbol{K}}$
$\mathcal{O}_d \leftarrow \widetilde{\mathrm{Dec}}_{\boldsymbol{K}}$
if $b = \mathtt{R}$:
$\mathcal{O}_e \leftarrow \$$
$\mathcal{O}_d \leftarrow \perp$
$\mathcal{A}$
$\vec{Q}^e_1, \ldots, \vec{Q}^e_{q_e}$
$\mathcal{O}_e(\vec{Q}^e_1), \ldots, \mathcal{O}_e(\vec{Q}^e_{q_e})$
$\vec{Q}^d_1, \ldots, \vec{Q}^d_{q_d}$
$\mathcal{O}_d(\vec{Q}^d_1), \ldots, \mathcal{O}_d(\vec{Q}^d_{q_d})$
b'

Fig. 11: *In the pseudo-random case, the encryption oracle $\mathcal{O}_e$ is the authenticated encryption scheme $\widetilde{\mathrm{Enc}}$ and the decryption oracle $\mathcal{O}_d$ is the authenticated decryption scheme $\widetilde{\mathrm{Dec}}$. In the truly random case, the encryption oracle always outputs a random ciphertext-tag pair and the decryption oracle always outputs $\perp$. The adversary $\mathcal{A}$ sends at most q_e encryption queries and at most q_d decryption queries to the corresponding oracles. At the end, the adversary outputs a flag guess b'.*

authenticated encryption scheme. Another restriction is, that $\mathcal{A}$ is not allowed to sent a decryption query $(\boldsymbol{A}_j, \boldsymbol{C}_j, N_j, T_j)$, where $\boldsymbol{C}_j, T_j$ were obtained by a prior encryption query $(\boldsymbol{A}_i, \boldsymbol{P}_i, N_i)$ with $\boldsymbol{A}_i = \boldsymbol{A}_j$ and $N_i = N_j$. This would lead to a trivial win, because $\mathcal{A}$ is then able to distinguish $\perp$ from $\widetilde{\mathrm{Dec}}$. This is also sound, because it does not make sense for an attacker in a real world scenario to decrypt a message, when he already knows the result. The advantage of $\mathcal{A}$ breaking $\widetilde{\Pi}$ in the nAE game is then defines as

$$\mathbf{adv}^{\mathrm{nAE}}_{\widetilde{\Pi}}(\mathcal{A}) = \left| \Pr\left[\mathcal{A}^{\$,\perp} \Rightarrow \mathtt{R}\right] - \Pr\left[\mathcal{A}^{\widetilde{\mathrm{Enc}},\widetilde{\mathrm{Dec}}} \Rightarrow \mathtt{R}\right] \right|.$$

This means, if $\mathcal{A}$ is not able to distinguish $\widetilde{\mathrm{Enc}}$ from \$ or $\widetilde{\mathrm{Dec}}$ from $\perp$ (with non-negligible probability), then the scheme is considered to be secure. This notion properly covers the security requirements for an authenticated encryption scheme. The indistinguishably of a ciphertext-tag pair from random values guarantees confidentiality. The inability of $\mathcal{A}$ to come up with a decryption query, that decrypts to a valid $\boldsymbol{P}$ instead of $\perp$ guarantees unforgeability and hence authenticity and integrity. Furthermore, the nonce-misuse-resistance property is taken into account by the game, because the adversary is allowed to repeat nonces (i.e. *nonce-disrespecting adversaries*), as long as the triples $(\boldsymbol{A}_i, \boldsymbol{P}_i, N_i)$ are distinct. The following main lemma relates the insecurity of the CCM-SIV scheme $\widetilde{\Pi}$ to the insecurities of the underlying building blocks Π and F^*. Based on the proof of this lemma, the proof of the main theorem (Theorem 1) will follow immediately.

Lemma 3. *[$\widetilde{\Pi}$'s security in the* nAE *game.] Let $\widetilde{\Pi}$ be the nAE scheme defined by Algorithm 8 and $\mathcal{A}$ be an adversary, who wins the* nAE *game defined in Figure 11 with non-negligible advantage. Then there exists a wrapping PPT adversary $\mathcal{B}$ who distinguishes F^* from a random function in the* PRF* *game defined in Figure 8 with non-negligible advantage and a wrapping PPT adversary $\mathcal{D}$ who distinguishes Π from its idealized version in the* ivE *game with non-negligible advantage. Particular, it holds that*

$$\mathbf{adv}_{\widetilde{\Pi}}^{\mathrm{nAE}}(\mathcal{A}) \leq \mathbf{adv}_{F^*}^{\mathrm{PRF}^*}(\mathcal{B}) + \mathbf{adv}_{\Pi}^{\mathrm{ivE}}(\mathcal{D}) + \frac{q_d}{2^x}. \tag{13}$$

Lemma 3 states, that $\widetilde{\Pi}$ must be a secure nAE scheme, because otherwise we would be able to break F^* and Π. This would be a contradiction to Lemma 2 and Definition 6, because if $\mathbf{adv}_{\widetilde{\Pi}}^{\mathrm{nAE}}(\mathcal{A})$ is non-negligible, either $\mathbf{adv}_{F^*}^{\mathrm{PRF}^*}(\mathcal{B})$ or $\mathbf{adv}_{\Pi}^{\mathrm{ivE}}(\mathcal{D})$ or both must be also non-negligible to fulfill (13).

Proof (Proof of Lemma 3.). The proof contains two reductions. The first reduction $\mathcal{B}$ is given in Algorithm 9 and the related framework is illustrated in Figure 12.

Algorithm 9 Pseudo Code for $\mathcal{B}$.

```
 1: $K_3 \xleftarrow{\$} \mathcal{K}$

 2: for all queries $\vec{Q}_i^e = (\boldsymbol{A}, \boldsymbol{P}, N)$ from $\mathcal{A}$ with $1 \leq i \leq q_e$ do
 3:     send $\vec{Q} = \vec{Q}_i^e$ to challenger
 4:     $T \leftarrow \mathcal{O}(\vec{Q})$
 5:     $\boldsymbol{C} \leftarrow$ CTR-$E_{K_3}^T(\boldsymbol{P})$
 6:     send $(\boldsymbol{C}, T)$ to $\mathcal{A}$

 7: for all queries $\vec{Q}_j^d = (\boldsymbol{A}, \boldsymbol{C}, N, T)$ from $\mathcal{A}$ with $1 \leq j \leq q_d$ do
 8:     $\boldsymbol{P} \leftarrow$ CTR-$E_{K_3}^T(\boldsymbol{C})$
 9:     send $\vec{Q} = (\boldsymbol{A}, \boldsymbol{P}, N)$ to challenger
10:     $T' \leftarrow \mathcal{O}(\vec{Q})$
11:     if $T = T'$ then
12:         send $\boldsymbol{P}$ to $\mathcal{A}$
13:     else
14:         send $\perp$ to $\mathcal{A}$

15: return same $b'$ as $\mathcal{A}$
```

The adversary $\mathcal{A}$ is playing the nAE game. She may output up to q_e encryption queries and up to q_d decryption queries in any order. After an encryption query, she expects the corresponding ciphertext-tag pair, and after a decryption query, she expects either the corresponding plaintext or the symbol $\perp$. The wrapping algorithm $\mathcal{B}$ has two handlers for these two types of $\mathcal{A}$-queries to simulate the authenticated encryption $\widetilde{\mathrm{Enc}}$ and the authenticated decryption $\widetilde{\mathrm{Dec}}$. Since $\mathcal{B}$ is playing the PRF* game, his goal is to distinguish between F^* and a random choice out of Funs$_{*,*,x\to x}$. For this, $\mathcal{B}$ outputs the same flag guess b' as $\mathcal{A}$ at the

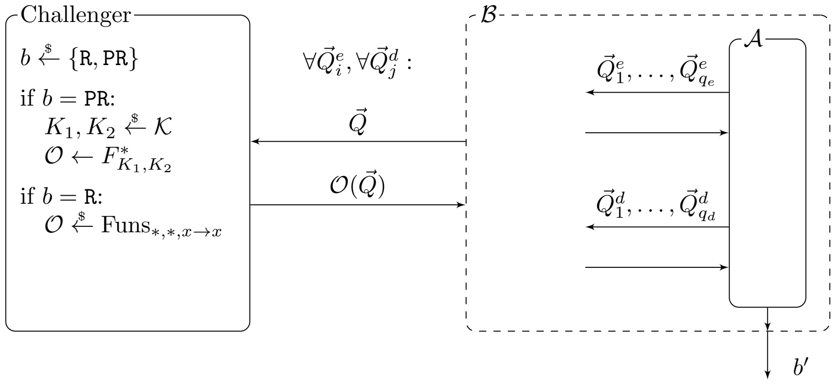

Challenger
$b \xleftarrow{\$} \{\mathtt{R}, \mathtt{PR}\}$
if $b = \mathtt{PR}$:
$K_1, K_2 \xleftarrow{\$} \mathcal{K}$
$\mathcal{O} \leftarrow F^*_{K_1,K_2}$
if $b = \mathtt{R}$:
$\mathcal{O} \xleftarrow{\$} \mathrm{Funs}_{*,*,x\to x}$
$\forall \vec{Q}^e_i, \forall \vec{Q}^d_j$:
$\vec{Q}$
$\mathcal{O}(\vec{Q})$
$\mathcal{B}$
$\mathcal{A}$
$\vec{Q}^e_1, \ldots, \vec{Q}^e_{q_e}$
$\vec{Q}^d_1, \ldots, \vec{Q}^d_{q_d}$
b'

Fig. 12: *The wrapping algorithm $\mathcal{B}$ tries to win the* PRF* *game on the outside by using the* nAE*-adversary $\mathcal{A}$ as a subroutine. $\mathcal{B}$ outputs the same flag guess b' as $\mathcal{A}$.*

end. In the beginning, $\mathcal{B}$ chooses the encryption key K_3 uniformly from the set $\mathcal{K}$. Upon receiving an encryption query $\vec{Q}^e_i$ from $\mathcal{A}$, he does the following: First, he passes the query $\vec{Q}^e_i$ directly to his challenger and uses the output $\mathcal{O}(\vec{Q}^e_i)$ as the tag T (as a replacement for the F^* evaluation). Then he runs the CTR-E encryption using the key K_3 and the tag T as IV. The resulting ciphertext $\boldsymbol{C}$ together with T is sent back to $\mathcal{A}$. Upon receiving a decryption query $\vec{Q}^d_j$ from $\mathcal{A}$, $\mathcal{B}$ does the following: First, he runs the CTR-E decryption using the key K_3 and the tag T as the IV, which is part of the decryption query, to obtain the unverified plaintext $\boldsymbol{P}$. Then he sends the triple $(\boldsymbol{A}, \boldsymbol{P}, N)$ to his challenger and uses the output as T' (as a replacement for the F^* evaluation). If $T = T'$, the unverified plaintext $\boldsymbol{P}$ is considered to be valid and sent back to $\mathcal{A}$. If $T \neq T'$, he sends back the symbol $\perp$. By definition, the advantages of $\mathcal{A}$ and $\mathcal{B}$ are

$$\mathbf{adv}^{\mathrm{nAE}}_{\widetilde{\Pi}}(\mathcal{A}) = \left| \Pr\left[\mathcal{A}^{\$,\perp} \Rightarrow \mathtt{R}\right] - \Pr\left[\mathcal{A}^{\widetilde{\mathrm{Enc}},\widetilde{\mathrm{Dec}}} \Rightarrow \mathtt{R}\right] \right|, \tag{14}$$

$$\mathbf{adv}^{\mathrm{PRF}^*}_{F^*}(\mathcal{B}) = \left| \Pr\left[\mathcal{B}^{\mathrm{Funs}_{*,*,x\to x}} \Rightarrow \mathtt{R}\right] - \Pr\left[\mathcal{B}^{F^*} \Rightarrow \mathtt{R}\right] \right|. \tag{15}$$

If $\mathcal{B}$'s challenger chooses $b = \mathtt{PR}$, then $\mathcal{B}$ perfectly simulates $\widetilde{\mathrm{Enc}}$ and $\widetilde{\mathrm{Dec}}$, because $\mathcal{O} = F^*$ in Algorithm 9. Since $\mathcal{B}$ outputs the same b' as $\mathcal{A}$, it holds that

$$\Pr\left[\mathcal{A}^{\widetilde{\mathrm{Enc}},\widetilde{\mathrm{Dec}}} \Rightarrow \mathtt{R}\right] = \Pr\left[\mathcal{B}^{F^*} \Rightarrow \mathtt{R}\right]. \tag{16}$$

Subtracting (15) from (14), applying the triangle inequality and making use of (16) yields

$$\mathbf{adv}^{\mathrm{nAE}}_{\widetilde{\Pi}}(\mathcal{A}) - \mathbf{adv}^{\mathrm{PRF}^*}_{F^*}(\mathcal{B}) \leq \left| \Pr\left[\mathcal{A}^{\$,\perp} \Rightarrow \mathtt{R}\right] - \Pr\left[\mathcal{B}^{\mathrm{Funs}_{*,*,x\to x}} \Rightarrow \mathtt{R}\right] \right|. \tag{17}$$

The second reduction $\mathcal{D}$ is trying to break the ivE-security of Π by running $\mathcal{A}$ as a subroutine.

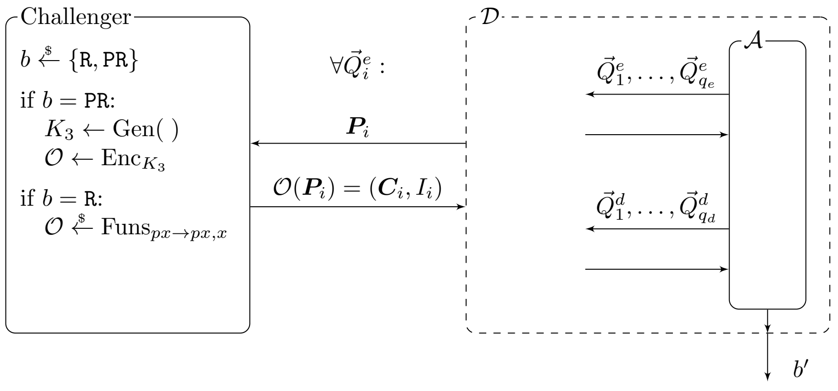
Algorithm 10 Pseudo Code for $\mathcal{D}$.

```
1: for all queries $\vec{Q}_i^e = (\boldsymbol{A}, \boldsymbol{P}, N)$ from $\mathcal{A}$ with $1 \le i \le q_e$ do
2:     send $\boldsymbol{P}_i = \boldsymbol{P}$ to challenger
3:     $(\boldsymbol{C}, T) \leftarrow \mathcal{O}(\boldsymbol{P}_i)$
4:     send $(\boldsymbol{C}, T)$ to $\mathcal{A}$

5: for all queries $\vec{Q}_j^d = (\boldsymbol{A}, \boldsymbol{C}, N, T)$ from $\mathcal{A}$ with $1 \le j \le q_d$ do
6:     send $\perp$ to $\mathcal{A}$

7: return same $b'$ as $\mathcal{A}$
```

The pseudo code for $\mathcal{D}$ is given in Algorithm 10 and the corresponding framework is illustrated in Figure 13. The adversary $\mathcal{D}$ plays the ivE game with his



Challenger
$b \xleftarrow{\$} \{\mathtt{R}, \mathtt{PR}\}$
if $b = \mathtt{PR}$:
$K_3 \leftarrow \mathrm{Gen}(\,)$
$\mathcal{O} \leftarrow \mathrm{Enc}_{K_3}$
if $b = \mathtt{R}$:
$\mathcal{O} \xleftarrow{\$} \mathrm{Funs}_{px \to px,x}$
$\forall \vec{Q}_i^e$:
$\boldsymbol{P}_i$
$\mathcal{O}(\boldsymbol{P}_i) = (\boldsymbol{C}_i, I_i)$
$\mathcal{D}$
$\mathcal{A}$
$\vec{Q}_1^e, \ldots, \vec{Q}_{q_e}^e$
$\vec{Q}_1^d, \ldots, \vec{Q}_{q_d}^d$
b'

Fig. 13: *The wrapping algorithm $\mathcal{D}$ tries to win the* ivE *game on the outside by using the* nAE-*adversary $\mathcal{A}$ as a subroutine. $\mathcal{D}$ outputs the same flag guess b' as $\mathcal{A}$.*

challenger and again consists of two handlers for the two different types of $\mathcal{A}$-queries. Upon receiving an encryption query $\vec{Q}_i^e$ from $\mathcal{A}$, $\mathcal{D}$ sends the plaintext $\boldsymbol{P}_i$ to his challenger as a replacement for $\widetilde{\mathrm{Enc}}$ and gets back the corresponding ciphertext-IV pair $(\boldsymbol{C}_i, I_i)$. $\mathcal{D}$ simply passes this pair to $\mathcal{A}$ and thereby misuses the IV I_i as the tag T. Upon receiving a decryption query $\vec{Q}_j^d$ from $\mathcal{A}$, $\mathcal{D}$ always answers with $\perp$. At the end, $\mathcal{D}$ outputs the same b' as $\mathcal{A}$. The advantage of $\mathcal{D}$ in the ivE game is defined as

$$\mathbf{adv}_{\Pi}^{\mathrm{ivE}}(\mathcal{D}) = \left| \Pr\left[\mathcal{D}^{\mathrm{Funs}_{px \to px,x}} \Rightarrow \mathtt{R}\right] - \Pr\left[\mathcal{D}^{\mathrm{Enc}} \Rightarrow \mathtt{R}\right] \right|. \tag{18}$$

Now consider the conditional probability space $b = \mathtt{R}$. By looking at Algorithm 10 and Figure 13, one can see that $\mathcal{A}$'s encryption queries are answered by a truly random function $\mathcal{O} \xleftarrow{\$} \text{Funs}_{px \to px, x}$ and $\mathcal{A}$'s decryption queries are always answered with $\perp$. Hence, $\mathcal{D}$ perfectly simulates $\$, \perp$. Therefore, it holds that

$$\Pr\left[\mathcal{A}^{\$,\perp} \Rightarrow \mathtt{R}\right] = \Pr\left[\mathcal{D}^{\text{Funs}_{px \to px, x}} \Rightarrow \mathtt{R}\right]. \tag{19}$$

Subtracting (18) from (17), applying the triangle equality and making use of (19) yields

$$\begin{aligned} &\mathbf{adv}^{\text{nAE}}_{\widetilde{\Pi}}(\mathcal{A}) - \mathbf{adv}^{\text{PRF}^*}_{F^*}(\mathcal{B}) - \mathbf{adv}^{\text{ivE}}_{\Pi}(\mathcal{D}) \\ &\leq \left| \Pr\left[\mathcal{D}^{\text{Enc}} \Rightarrow \mathtt{R}\right] - \Pr\left[\mathcal{B}^{\text{Funs}_{*,*,x \to x}} \Rightarrow \mathtt{R}\right] \right|. \end{aligned} \tag{20}$$

It remains to bound the quantity on the right side of (20). For this, we have to analyze, under which circumstances $\mathcal{D}^{\text{Enc}}$ behaves different from $\mathcal{B}^{\text{Funs}_{*,*,x \to x}}$. Let us first compare the encryption query handlers in Algorithm 10 and 9. Since $\mathcal{D}$ is interacting with Enc, an encryption query $\vec{Q}^e_i = (\boldsymbol{A}, \boldsymbol{P}, N)$ from $\mathcal{A}$ is answered with $(\boldsymbol{C}, T) \leftarrow \text{Enc}_{K_3}(\boldsymbol{P})$. Since $\mathcal{B}$ is interacting with $\text{Funs}_{*,*,x \to x}$, an encryption query $\vec{Q}^e_i = (\boldsymbol{A}, \boldsymbol{P}, N)$ from $\mathcal{A}$ is answered with $(\boldsymbol{C}, T)$, where T is chosen uniformly by $\mathcal{O} \xleftarrow{\$} \text{Funs}_{*,*,x \to x}$ and $\boldsymbol{C} \leftarrow \text{CTR-}E^T_{K_3}(\boldsymbol{P})$. But this is exactly $\text{Enc}_{K_3}(\boldsymbol{P})$. Therefore, both $\mathcal{D}$ and $\mathcal{B}$ answer encryption queries from $\mathcal{A}$ with the same probability distribution. Now we have to compare the decryption handlers in Algorithm 10 and 9. $\mathcal{D}$ answers all decryption queries $\vec{Q}^d_j$ from $\mathcal{A}$ with $\perp$, whereas $\mathcal{B}$ answers with the plaintext $\boldsymbol{P}$ instead of $\perp$, if the two tags T and T' accidentally match up. T is chosen by the adversary $\mathcal{A}$, whereas T' is calculated by $\mathcal{O} \xleftarrow{\$} \text{Funs}_{*,*,x \to x}$. Let match denote the event, that at least one such a match occurs during the decryption queries. Then conditioned to the case $\overline{\text{match}}$, $\mathcal{D}$ always outputs $\perp$ and hence $\mathcal{D}^{\text{Enc}}$ behaves the same as $\mathcal{B}^{\text{Funs}_{*,*,x \to x}}$. With this, we can rewrite the right side of (20):

$$\begin{aligned} &\left| \Pr\left[\mathcal{D}^{\text{Enc}} \Rightarrow \mathtt{R}\right] - \Pr\left[\mathcal{B}^{\text{Funs}_{*,*,x \to x}} \Rightarrow \mathtt{R}\right] \right| \\ &= \Big| \Pr\left[\mathcal{D}^{\text{Enc}} \Rightarrow \mathtt{R} \mid \text{match}\right] \Pr[\text{match}] + \Pr\left[\mathcal{D}^{\text{Enc}} \Rightarrow \mathtt{R} \mid \overline{\text{match}}\right] \Pr\left[\overline{\text{match}}\right] \\ &- \Pr\left[\mathcal{B}^{\text{Funs}_{*,*,x \to x}} \Rightarrow \mathtt{R} \mid \text{match}\right] \Pr[\text{match}] \\ &- \Pr\left[\mathcal{B}^{\text{Funs}_{*,*,x \to x}} \Rightarrow \mathtt{R} \mid \overline{\text{match}}\right] \Pr\left[\overline{\text{match}}\right] \Big| \\ &= \Pr[\text{match}] \left| \Pr\left[\mathcal{D}^{\text{Enc}} \Rightarrow \mathtt{R} \mid \text{match}\right] - \Pr\left[\mathcal{B}^{\text{Funs}_{*,*,x \to x}} \Rightarrow \mathtt{R} \mid \text{match}\right] \right| \\ &\leq \Pr[\text{match}]. \end{aligned} \tag{21}$$

The tag T' is uniformly distributed and hence takes the value T with probability $1/2^x$. For q_d distinct decryption queries the union bound gives

$$\Pr[\text{match}] \leq \frac{q_d}{2^x}. \tag{22}$$

Combining (20), (21) and (22) proves the lemma:

$$\mathbf{adv}_{\widetilde{\Pi}}^{\mathrm{nAE}}(\mathcal{A}) - \mathbf{adv}_{F^*}^{\mathrm{PRF}^*}(\mathcal{B}) - \mathbf{adv}_{\Pi}^{\mathrm{ivE}}(\mathcal{D}) \leq \frac{q_d}{2^x}.$$

□

Lemma 3 gives an upper bound for the nAE-security of $\widetilde{\Pi}$ by relating it with the PRF*- and ivE-security of the underlying building blocks F^* and Π. The main theorem of the paper (Theorem 1) gives an upper bound for the nAE-security of $\widetilde{\Pi}$ by relating it with the PRF-security of the underlying PRF E. We now have everything together to do the proof for Theorem 1 in a straightforward way:

Proof (Proof of Theorem 1). Let $\mathcal{B}$ be an adversary incorporating the PRF-adversary from Lemma 2 and the PRF-adversary from Definition 6. Then substituting the advantages in Lemma 3 with the bounds in Lemma 2 and Definition 6 yields

$$\mathbf{adv}_{\widetilde{\Pi}}^{\mathrm{nAE}}(\mathcal{A}) \leq 2 \cdot \mathbf{adv}_{E}^{\mathrm{PRF}}(\mathcal{B}) + \epsilon \cdot \frac{(q_e + q_d)^2}{2} + \frac{q_e^2 \cdot p_{\max}}{2 \cdot 2^x} + \frac{q_d}{2^x}. \tag{23}$$

□

The maximum number of allowed plaintext blocks $p_{\max}$ in Theorem 1 is dependent on the bit length of the counter field in the CTR mode encryption. For a c bit counter (in case of CCM-SIV $c = 32$ bit), $p_{\max} = 2^c$. So the dominating term in (23) is

$$\frac{q_e^2 \cdot p_{\max}}{2 \cdot 2^x} = \frac{q_e^2}{2 \cdot 2^{x-c}}.$$

5 Differences to GCM-SIV and Nonce-based Key Derivation

Our main theorem (Theorem 1) differs from [7, Theorem 4.2] of the 3-key instantiation of the GCM-SIV scheme, where $p_{\max} = 2^k - 1$, the bit length of the counter is represented as k (instead of c) and the block size is represented as n (instead of x), by a factor 1/2 in two terms. This is due to the fact that we upper bound $\sum_{i=0}^{q-1} i$ by $q^2/2$ instead of q^2.

A modification to a 2-key variant of CCM-SIV is straightforward. Furthermore, an extension to a nonce-based key derivation as proposed in [6, Fig. 2] and [11, Section 5] to reduce the dominating term $q_e^2 \cdot p_{\max}/2 \cdot 2^x$ in Theorem 1 to $QR^2 \cdot p_{\max}/2 \cdot 2^x$ (using the PRF-advantage), where Q is the number of distinct nonces used throughout all encryption queries and R is the maximal number of repetitions of any nonce in all encryption queries, is realizable.

6 Conclusion

We introduced a new nonce-misuse-resistant authenticated encryption scheme called CCM-SIV, which only uses a single type of PRF as the underlying cryptographic primitive (AES). This makes the scheme compatible with most existing AES software libraries and hardware accelerators. Since we use three independent subkeys, which can be derived from a single master key by a deterministic pseudo random number generator, no strange internal collisions on the inputs of the AES block cipher can occur between the MAC generation and the symmetric encryption parts. Therefore, no modifications have to be applied to circumvent possible internal collisions. This makes the scheme cryptographically strong and easy to implement in both hardware and software.

References

1. Bailey, D., McGrew, D.: AES-CCM Cipher Suites for Transport Layer Security (TLS) (2018), `https://tools.ietf.org/html/rfc6655`
2. Boeck, H., Zauner, A., Devlin, S., Somorovsky, J., Jovanovic, P.: Nonce-disrespecting Adversaries: Practical Forgery Attacks on GCM in TLS. In: Proceedings of the 10th USENIX Conference on Offensive Technologies. pp. 15–25 (2016), `http://dl.acm.org/citation.cfm?id=3027019.3027021`
3. Boneh, D., Shoup, V.: A Graduate Course in Applied Cryptography. Book Draft v0.3 (2016)
4. Dworkin, M.J.: SP 800-38C. Recommendation for Block Cipher Modes of Operation: The CCM Mode for Authentication and Confidentiality. Tech. rep., NIST (2004)
5. Gueron, S.: Intel® Advanced Encryption Standard (Intel® AES) Instructions Set - Rev (2012)
6. Gueron, S., Langley, A., Lindell, Y.: AES-GCM-SIV: Specification and Analysis. IACR Cryptology ePrint Archive (2017)
7. Gueron, S., Lindell, Y.: GCM-SIV: Full Nonce Misuse-Resistant Authenticated Encryption at Under One Cycle Per Byte. In: Proceedings of the 22^{nd} ACM SIGSAC Conference on Computer and Communications Security. pp. 109–119. ACM (2015), `http://doi.acm.org/10.1145/2810103.2813613`
8. Gueron, S., Lindell, Y.: Better Bounds for Block Cipher Modes of Operation via Nonce-Based Key Derivation. Tech. Rep. 702 (2017), `https://eprint.iacr.org/2017/702`
9. Housley, R.: Using Advanced Encryption Standard (AES) CCM Mode with IPsec Encapsulating Security Payload (ESP) (2018), `https://tools.ietf.org/html/rfc4309`
10. Iwata, T., Minematsu, K.: Stronger Security Variants of GCM-SIV. IACR Transactions on Symmetric Cryptology 2016(1), 134–157 (Dec 2016), `https://tosc.iacr.org/index.php/ToSC/article/view/539`
11. Iwata, T., Seurin, Y.: Reconsidering the Security Bound of AES-GCM-SIV. Tech. Rep. 708 (2017), `https://eprint.iacr.org/2017/708`

12. Joux, A.: Authentication failures in NIST version of GCM (Jan 2006)
13. Kresmer, P.: CCM-SIV, Reference C Implementation (2019), https://gitlab.com/pytrack/ccm-siv
14. Lindell, Y., Langley, A., Gueron, S.: AES-GCM-SIV: Nonce Misuse-Resistant Authenticated Encryption (2018), https://tools.ietf.org/html/draft-gueron-gcmsiv-00
15. McGrew, D., Viega, J.: The Galois/Counter Mode of Operation (GCM) (2004)
16. Namprempre, C., Rogaway, P., Shrimpton, T.: Reconsidering Generic Composition. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 257–274. Springer (2014)
17. Poledna, S.: Fault-Tolerant Real-Time Systems: The Problem Of Replica Determinism. Springer (2013)
18. Rogaway, P., Shrimpton, T.: Deterministic Authenticated-Encryption: A Provable-Security Treatment of the Key-Wrap Problem. In: Advances in Cryptology–EUROCRYPT. vol. 6 (2007)
19. Rogaway, P., Shrimpton, T.: A Provable-Security Treatment of the Key-Wrap Problem. In: Advances in Cryptology - EUROCRYPT 2006. pp. 373–390. Lecture Notes in Computer Science, Springer, Berlin, Heidelberg (May 2006), https://link.springer.com/chapter/10.1007/11761679_23
20. Whiting, D., Housley, R., Ferguson, N.: Counter with CBC-MAC (CCM). RFC 3610 (September 2003)