How to Securely Release Unverified Plaintext in Authenticated Encryption

Elena Andreeva[1,2], Andrey Bogdanov[3], Atul Luykx[1,2], Bart Mennink[1,2], Nicky Mouha[1,2], and Kan Yasuda[1,4]

[1] Department of Electrical Engineering, ESAT/COSIC, KU Leuven, Belgium.
firstname.lastname@esat.kuleuven.be
[2] iMinds, Belgium.
[3] Department of Mathematics, Technical University of Denmark, Denmark.
anbog@dtu.dk
[4] NTT Secure Platform Laboratories, Japan.
yasuda.kan@lab.ntt.co.jp

Abstract. We consider the case where an authenticated encryption scheme outputs the decrypted plaintext before successful verification. This scenario raises many security issues, and is highlighted in the upcoming CAESAR competition. It arises for example when devices have insufficient memory to store the entire plaintext, or when the decrypted plaintext needs to be processed early due to real-time requirements. Firstly, we formalize the *releasing unverified plaintext* (RUP) setting. To achieve privacy in this setting, we propose using *plaintext awareness* (PA) along with IND-CPA. An authenticated encryption scheme is PA if there exists a *plaintext extractor* for every adversary. The plaintext extractor does not know the secret key, but tries to fool the adversary by mimicking the decryption oracle. The release of unverified plaintext then becomes harmless, because it is infeasible to distinguish between answers from the real decryption oracle and from the plaintext extractor. We introduce two notions of plaintext awareness in the symmetric-key setting (PA1 and PA2), and show implications and separations between PA1, PA2, and existing notions. To achieve integrity of the ciphertexts, INT-CTXT in the RUP setting is required, which we refer to as INT-RUP. These security notions are then used to make a classification of symmetric-key schemes in the RUP setting. We analyze existing authenticated encryption schemes in this setting, and provide solutions to fix insecure schemes.

Keywords. Symmetric-key Cryptography, Authenticated Encryption, Releasing Unverified Plaintext, Plaintext Awareness, Plaintext Extractor, CAESAR Competition.

1 Introduction

The goal of authenticated encryption (AE) is to provide both privacy and integrity. The decryption of AE conventionally consists of two phases: *decryption* and *verification*. As reflected in classical security models, the plaintext coming from the decryption is released only upon successful verification.

However, in certain settings it may be desirable to release the plaintext before verification is complete. It is necessary to do so if there is not enough memory to store the entire plaintext [24] or because real-time requirements would otherwise not be met [17, 43]. Even beyond these settings, releasing unverified plaintext allows us to increase the efficiency of certain applications. For example, let us consider a device with insecure memory [42]. If we want to completely avoid releasing unverified plaintext into this insecure memory, we can use Encrypt-then-MAC in two passes: a first pass to verify the MAC, and a second pass to decrypt the ciphertext. However, we can replace this by an AE scheme that uses only a single pass if this AE scheme is secure against the release of unverified plaintext.

If the attacker cannot observe the unverified plaintext directly, it may be possible to determine properties of the plaintext through a side channel. This occurs, for example, in the padding oracle attacks introduced by Vaudenay [44], where an error message or the lack of an acknowledgment indicates whether the unverified plaintext was correctly padded. Canvel et al. [20] showed how to mount a padding oracle attack on the then-current version of OpenSSL

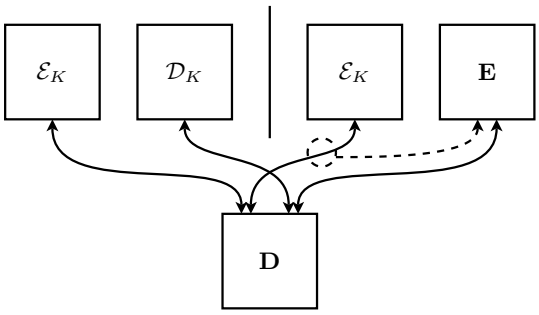

Fig. 1. The two plaintext aware settings (PA1 and PA2) used in the paper, where **D** is an adversary. Not shown in the figure is the type of IV used by the $\mathcal{E}_K$ oracle (cf. Sect. 3.2). **Left:** Real world, with encryption oracle $\mathcal{E}_K$ and decryption oracle $\mathcal{D}_K$. **Right:** Simulated world, with encryption oracle $\mathcal{E}_K$ and plaintext extractor **E**. The plaintext extractor **E** is a stateful algorithm without knowledge of the secret key K, nor access to the encryption oracle $\mathcal{E}_K$. As shown by the dotted line, **E** has access to the encryption queries made by adversary **D** in the PA1 setting, but not in the PA2 setting.

by exploiting timing differences in the decryption processing of TLS. As shown by Paterson and AlFardan [1, 34] for TLS and DTLS, it is very difficult to prevent that the attacker can learn the cause of decryption failures.

The issue of releasing unverified plaintext has also been acknowledged and explicitly discussed in the upcoming CAESAR competition [15]: *"Beware that security questions are raised by any authenticated cipher that handles a long ciphertext in one pass without using a large buffer: releasing unverified plaintext to applications often means releasing it to attackers and also requires an analysis of how the applications will react."*

For several AE schemes, including OCB [30], AEGIS [46], ALE [17], and FIDES [17], the designers explicitly stress that unverified plaintext cannot be released. Although the issue of *releasing unverified plaintext* (RUP) in AE is frequently discussed in the literature, it has remained unaddressed even in recent AE proposals. This is likely due to the lack of comprehensive study.

We would like to mention explicitly that we certainly do not recommend omitting verification. Verification is an essential part of an AE scheme, and is necessary to prevent that incorrect plaintexts are accepted anyway. However, our scenario assumes that the attacker can get hold of the unverified plaintext, or any information relating to it, before verification is complete.

1.1 Security Under Release of Unverified Plaintext

The main goal of this work is to formalize security for authenticated encryption with the release of unverified plaintext. To achieve integrity in the RUP setting, we introduce the notion INT-RUP. For privacy, we propose using both IND-CPA and *plaintext awareness* (PA).

INT-RUP. The goal of an adversary under INT-CTXT is to produce AE ciphertexts which pass the verification phase of the decryption. In addition, the adversary can make encryption queries. We translate the integrity of ciphertexts INT-CTXT notion into the RUP setting, which we call INT-RUP.

What the new notion of INT-RUP allows, in addition to INT-CTXT, is giving the adversary the ability to do decryption queries and observe the unverified plaintexts. Notice that to allow this we require an alternative AE syntax that explicitly separates the decryption and verification functionalities.

Plaintext Awareness (PA). In the RUP setting an adversary can observe the unverified plaintexts resulting from decryption queries. We introduce PA as a new symmetric-key notion

to achieve security in this setting. Informally, we define a scheme to be PA if the adversary cannot gain any additional knowledge about the plaintext from decryption queries besides what it can derive from encryption queries.

Our PA notion involves the encryption and decryption functionalities, and can thus be defined both for encryption schemes, as well as for AE schemes that release unverified plaintext.

At the heart of our new PA notion is the *plaintext extractor*, shown in Fig. 1. We say that an encryption scheme is PA if there exists an efficient plaintext extractor for every adversary. The plaintext extractor is a stateful algorithm with the goal of mimicking the decryption oracle in order to fool the adversary. It cannot make encryption nor decryption queries, and does not know the secret key. We define two notions of plaintext awareness: PA1 and PA2. The extractor is given access to the history of queries made to the encryption oracle in PA1, but not in PA2. Hence PA1 is used to model RUP scenarios in which the goal of the adversary is to gain knowledge beyond what it knows from the query history. For situations in which the goal of the adversary is to decrypt one of the ciphertexts in the query history, we require PA2.

Relations Among Notions. Bellare and Rogaway [11] introduced the notion of PA for public-key encryption. PA for public-key encryption without random oracles was defined by Bellare and Palacio [10]. In the symmetric-key setting, our definition of PA is somewhat similar, however there are important technical differences which make the public-key results inapplicable to the symmetric-key setting.

The relations among the PA notions and the conventional security notions for encryption (see Sect. 3.3) are summarized in Fig. 2. We consider three different IV assumptions: random IV, nonce IV (non-repeating value), and arbitrary IV (value that can be reused), as explained in Sect. 3.2. The statements of the theorems and proofs can be found in Sect. 5.

The motivation for having two separate notions, PA1 and PA2, is as follows. As we prove in this work, if the plaintext extractor has access to the query history (PA1), then there are no implications between IND-CPA+PA1 and IND-CCA. However, if we modify plaintext awareness so that the plaintext extractor no longer has access to the query history (PA2), then we can prove that IND-CPA+PA2 implies IND-CCA$'$. IND-CCA$'$ is a strengthened version of IND-CCA, where we allow the adversary to re-encrypt the outputs of the decryption oracle. Note that such a re-encryption would always be allowed in the public-key setting, but not in the symmetric-key setting where the key required for encryption is secret.

Furthermore, we also prove that PA2 is equivalent to the notion of *decryption independence* (DI). DI captures the fact that encryption and decryption under the same key are only related to each other as much as encryption and decryption under different keys.

Finally, INT-RUP clearly implies INT-CTXT. The opposite is, however, not necessarily true.

Motivating Examples. To get an intuition of PA1 (shown in Fig. 1) and how it relates to the RUP setting, we provide two motivating examples with CTR mode. For simplicity, here we define the encryption function of CTR mode as $\mathcal{E}_K(\mathrm{IV}, P) = E_K(\mathrm{IV}) \oplus M$, where the message M and the initialization value IV consist of one block each, and E_K is a block cipher with a secret key K. The corresponding decryption function is $\mathcal{D}_K(\mathrm{IV}, C) = E_K(\mathrm{IV}) \oplus C$. As shown in [13], CTR mode is IND-CPA but not IND-CCA, a result that holds for nonce IVs (unique non-repeating values) as well as for random IVs.

1. *CTR mode with a nonce IV is not PA1.* Following Rogaway [35], we assume that an adversary is free to specify the IV for encryption and decryption queries, as long as it does not make two encryption queries with the same nonce IV. In the attack, an adversary first makes a decryption query (N, C) with nonce N and one-block ciphertext C to obtain a message M. The correct decryption of M is $E_K(N) \oplus C$ as output by the decryption oracle. The

adversary then computes the keystream $\kappa := M \oplus C$. Now in a second query (N, M'), this time to the encryption oracle, the adversary obtains C' where $C' = M' \oplus \kappa$.

Plaintext awareness fails in this case: the reply that a plaintext extractor would give for the decryption query, cannot be consistent with a high probability for the encryption query that follows it. That is because the plaintext extractor cannot compute κ at the time of the first decryption query for the following reasons: (i) it does not know the secret key K; (ii) it is not allowed to do encryption queries; (iii) an encryption query with N has not yet been recorded in the query history.

2. *CTR mode with a random IV is PA1.* In this setting, the IV used in encryption is chosen randomly by the environment, and therefore outside of the control of the attacker. However, the adversary can still freely choose the IV for its decryption queries. In this random IV setting, the attack in the nonce IV example does not apply. To see this, let us consider the situation where an adversary queries the decryption of (IV_1, C) with a one-block ciphertext C. The adversary can compute the keystream associated to IV_1, but it does not control when the same IV_1 will be used in encryption. Thus, a plaintext extractor can be defined as just outputting a random plaintext M in response to the (IV_1, C) query.

   But what if an adversary makes further decryption queries with the same IV? Suppose the adversary makes decryption query $(\mathrm{IV}_1, C \oplus \Delta)$. Since the plaintext extractor is a stateful algorithm, it can simply output $M \oplus \Delta$ to provide consistency. Furthermore, if an adversary makes encryption queries, these will be seen by the PA1 plaintext extractor. Therefore, the plaintext extractor can calculate the keystream from these queries, and respond to any decryption queries in a consistent way. A full proof for the PA1 security of CTR mode with random IVs is provided in Prop. 2.

AE schemes such as GCM [31] and CCM [45] reduce to CTR mode in the RUP setting. This is because the adversary does not need to forge a ciphertext in order to obtain information about the corresponding (unverified) plaintext. By requiring that the underlying encryption scheme of an AE scheme is PA1, we ensure that the adversary does not gain any information from decryption queries, in the sense that no decryption query can be used to find an inconsistency with any past or future queries to the encryption or decryption oracles.

1.2 Background and Further Related Work

In 2004, Rogaway [36] formalized the notion of encryption schemes based on a nonce IV, in contrast with prior encryption modes that used a random IV (as in the CBC mode standardized by NIST in 1980 [33]).

Rogaway and Shrimpton [38] formalized the notion of deterministic AE (DAE). In DAE, an IV input is optional and can therefore take arbitrary values. A secure DAE differs from a secure nonce IV AE scheme in the fact that the DAE privacy is possible only up to message repetitions, namely an adversary can detect repetitions of encryptions of identical messages. Unfortunately, the encryption of DAE schemes is not online. To resolve this issue, Fleischmann et al. [23] introduced the notion of authenticated online encryption. The syntax here is the same as DAE and privacy holds only up to repetitions of messages with identical prefixes or up to the longest common prefix.

Also, the notion of AE has been extended and generalized in different ways. Tsang et al. [43] gave syntax and security definitions of AE for streaming data. Bellare and Keelveedhi [7] considered a stronger security model where data may be key-dependent. Boldyreva et al. reformulated AE requirements and properties to handle ciphertext fragmentation in [18], and enhanced the syntax and security definitions so that the verification oracle is allowed to handle multiple failure events in [19].

Table 1. PA1 and PA2 security of some deterministic and non-deterministic schemes. In the columns for PA1 and PA2, ✓ means secure (there exists an extractor), and ✗ means insecure (there exists an attack). Proofs for the security results in this table can be found in Sect. 6.

IV type	Online	Scheme	PA1	PA2	Remark
random	✓	CTR, CBC [33]	✓	✗	
nonce	✓	OCB [37]	✗	✗	
	✓	GCM [31], SpongeWrap [16]	✗	✗	
	✗	CCM [45]	✗	✗	not online [39]
arbitrary	✓	COPA [3]	✗	✗	privacy up to prefix
	✓	McOE-G [23]	✗	✗	〃
	✓	APE [2]	✓	✗	〃, backwards decryption
	✗	SIV [38], BTM [27], HBS [28]	✓	✗	privacy up to repetition
	✗	Encode-then-Encipher [12]	✓	✓	〃, VIL SPRP, padding

1.3 Analysis of Authenticated Encryption Schemes

We categorize existing AE schemes based on the type of IV used by the encryption function: random IV, nonce IV, and arbitrary IV. The decryption function is assumed to be deterministic and stateless; we assume that the adversary can specify any IV value to the decryption oracle.

We analyze the security in the RUP setting of several recently proposed AE schemes (BTM [27], HBS [28], SpongeWrap [16], McOE-G [23], APE [2], and COPA [3]), as well as the more established AE schemes (Encode-then-Encipher [12], OCB [37], CCM [45], GCM [31], and SIV [38]). An overview of our privacy results is provided in Table 1, where we additionally include the encryption-only modes CTR and CBC as random IV examples. We draw a distinction between the schemes that are *online* and the schemes that are not. An online (encryption) scheme is able to produce the ciphertext block as it receives a plaintext block.

Most of the schemes in Table 1 fail to satisfy the notion of PA1 security. To this end, we propose several techniques to restore PA1 for deterministic schemes, which fall in the arbitrary IV class, and nonce-based schemes. Firstly, for nonce-based schemes, we introduce the *nonce decoy* technique. Next, for the arbitrary IV setting we propose the PRF-to-IV method which converts a random IV PA1 scheme into an arbitrary IV PA1 scheme. For online deterministic AE schemes, we demonstrate that PA1 security can be achieved only if the ciphertext is substantially longer than the plaintext, or the decryption is offline. We show that McOE-G [23] achieves PA1 if the plaintext is padded so that the ciphertext becomes twice as long. We also prove that APE [2], an online deterministic AE scheme with offline decryption, achieves PA1.

For OCB [37] and COPA [3], we show how to violate the INT-RUP security by using the unverified plaintext to construct forgeries for these schemes. Finally we show that the nonce decoy preserves INT-RUP, and the PRF-to-IV method turns any random IV scheme into an INT-RUP arbitrary IV scheme.

2 Preliminaries

Symbols. Given two strings A and B in $\{0,1\}^*$, we use $A\|B$ and AB interchangeably to denote the concatenation of A and B. The symbol $\oplus$ denotes the bitwise XOR operation of two (or more) strings. The addition $+$ is performed modulo 2^n, where n usually is the bit length of a block. For example, in the CTR mode of operation of a block cipher, we increment the IV value by addition $IV + i \pmod{2^n}$, where n is the block size, the n-bit string $IV = IV_{n-1} \cdots IV_1 IV_0 \in \{0,1\}^n$ is converted to an integer $2^{n-1}IV_{n-1} + \cdots + 2IV_1 + IV_0 \in \{0, 1, \ldots, 2^n - 1\}$, and the result of addition is converted to an n-bit string in the reverse way. By $K \xleftarrow{R} \mathsf{K}$ we mean that K is chosen uniformly at random from the set K.

Adversaries and Advantages. An adversary is an oracle Turing machine. Let $\mathbb{D}$ be some class of computationally bounded adversaries; a class $\mathbb{D}$ can consist of a single adversary $\mathbf{D}$, i.e. $\mathbb{D} = \{\mathbf{D}\}$, in which case we write simply $\mathbf{D}$ instead of $\mathbb{D}$. For convenience, we use the notation

$$\underset{\mathbb{D}}{\Delta}(f\,;\,g) := \sup_{\mathbf{D}\in\mathbb{D}} \Big|\Pr[\mathbf{D}^f = 1] - \Pr[\mathbf{D}^g = 1]\Big|$$

to denote the supremum of the distinguishing advantages over all adversaries distinguishing oracles f and g, where the notation $\mathbf{D}^{\mathcal{O}}$ indicates the value output by $\mathbf{D}$ after interacting with oracle $\mathcal{O}$. The probabilities are defined over the random coins used in the oracles and the random coins of the adversary, if any. Multiple oracles are separated by a comma, e.g. $\Delta(f_1, f_2\,;\,g_1, g_2)$ denotes distinguishing the combination of f_1 and f_2 from the combination of g_1 and g_2.

If $\mathbf{D}$ is distinguishing $(f_1, f_2, \ldots, f_k)$ from $(g_1, g_2, \ldots, g_k)$, then by $\mathcal{O}_i$ we mean the ith oracle that $\mathbf{D}$ has access to, i.e. either f_i or g_i depending upon which oracles it is interacting with. With $\mathcal{O}_i \hookrightarrow \mathcal{O}_j$ we are describing an action that $\mathbf{D}$ performs: first $\mathbf{D}$ queries $\mathcal{O}_i$, and then at some point in the future $\mathbf{D}$ queries $\mathcal{O}_j$ with the output of $\mathcal{O}_i$ (assuming it makes sense to use the output of $\mathcal{O}_i$ as the input for $\mathcal{O}_j$ directly).

Pseudo-Random Function (PRF). Let $G : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^m$ be a function and let $\mathbf{D}$ be an adversary accessing one oracle. The PRF advantage of $\mathbf{D}$ with respect to G is defined as

$$\mathsf{PRF}_G(\mathbf{D}) := \underset{\mathbf{D}}{\Delta}(G_K\,;\,\Phi)\ ,$$

where $\Phi : \{0,1\}^n \to \{0,1\}^m$ is a uniform random function and $K \xleftarrow{R} \{0,1\}^n$. We call G a PRF if the PRF advantage over all $\mathbf{D}$ is "small" (we do not need to define what small is for our purposes). For a variable-input-length (VIL) PRF $G : \{0,1\}^k \times \{0,1\}^* \to \{0,1\}^m$ the PRF advantage is defined analogously.

Strong Pseudo-Random Permutation (SPRP). Let $E : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^m$ be a block cipher and let $\mathbf{D}$ be an adversary accessing one oracle. The SPRP advantage of $\mathbf{D}$ with respect to G is defined as

$$\mathsf{SPRP}_E(\mathbf{D}) := \underset{\mathbf{D}}{\Delta}(E_K, D_K\,;\,\Phi, \Phi^{-1})\ ,$$

where $\Phi : \{0,1\}^n \to \{0,1\}^m$ is a uniform random permutation and $K \xleftarrow{R} \{0,1\}^n$. We call E an SPRP if the SPRP advantage over all $\mathbf{D}$ is "small" (we do not need to define what small is for our purposes).

Online Functions. A function $f : \mathsf{M} \to \mathsf{C}$ is said to be *n-online* if there exist functions $f_i : \{0,1\}^i \to \{0,1\}^{c_i}$ and $f'_i : \{0,1\}^i \to \{0,1\}^{c'_i}$ such that $c_i > 0$, and for all $M \in \mathsf{M}$ we have

$$f(M) = f_n(M_1)\, f_{2n}(M_1M_2) \cdots f_{jn}(M_1M_2\cdots M_j)\, f'_{|M|}(M)\ ,$$

where $j = \lfloor(|M| - 1)/n\rfloor$ and M_i is the ith n-bit block of M. Often we just say f is *online* if the value n is clear from the context.

3 AE Schemes: Syntax, Types, and Security

3.1 New AE Syntax

A conventional AE scheme $\Pi = (\mathcal{E}, \mathcal{D})$ consists of an encryption algorithm $\mathcal{E}$ and a decryption algorithm $\mathcal{D}$, where we write

$$(C, T) \leftarrow \mathcal{E}_K(IV, A, M)\ ,$$
$$M/\perp \leftarrow \mathcal{D}_K(IV, A, C, T)\ ,$$

where $K \in \mathsf{K}$ is a key, $IV \in \mathsf{IV}$ an initialization value IV, $A \in \mathsf{A}$ associated data, $M \in \mathsf{M}$ a message, $C \in \mathsf{C}$ the ciphertext, and $T \in \mathsf{T}$ the tag. Each of these sets is a subset of $\{0,1\}^*$. The *correctness condition* states that for all K and IV, if $\mathcal{E}_K(IV, A, M) = (C, T)$, then $\mathcal{D}_K(IV, A, C, T) = M$. A secure AE scheme should return $\perp$ when it does not receive a valid (C, T) pair.

In order to consider what happens when unverified plaintext is released, we must disconnect the decryption algorithm from the verification algorithm so that the decryption algorithm always releases plaintext. A separated AE scheme is a triplet $\Pi = (\mathcal{E}, \mathcal{D}, \mathcal{V})$ of keyed algorithms — encryption $\mathcal{E}$, decryption $\mathcal{D}$, and verification $\mathcal{V}$ — such that

$$\begin{aligned} (IV, A, C, T) &\leftarrow \mathcal{E}_K(IV, A, M) \ , \\ (IV, A, M) &\leftarrow \mathcal{D}_K(IV, A, C, T) \ , \\ \top/\perp &\leftarrow \mathcal{V}_K(IV, A, C, T) \ , \end{aligned}$$

where K, IV, A, M, C, and T are defined as above. For convenience in stating the security definitions later on, we have defined the encryption and decryption algorithms so that the IV and A are also output, but if IV and A are not explicitly mentioned as outputs, then we implicitly include them. Note that in some deterministic schemes IV may be absent. Furthermore, for simplicity we might omit A if there is no associated data. The special symbols $\top$ and $\perp$ indicate the success and failure of the verification process, respectively.

As in the conventional setting we require a *correctness condition*: for all K and IV such that $\mathcal{E}_K(IV, A, M) = (IV, A, C, T)$, we require $\mathcal{D}_K(IV, A, C, T) = (IV, A, M)$ and $\mathcal{V}_K(IV, A, C, T) = \top$.

Relation to Conventional Syntax. Given a separated AE scheme $\Pi = (\mathcal{E}, \mathcal{D}, \mathcal{V})$, we can easily convert it into a conventional AE scheme $\overline{\Pi} = (\mathcal{E}, \overline{\mathcal{D}})$. Remember that the conventional decryption oracle $\overline{\mathcal{D}}_K(IV, A, C, T)$ outputs M where $(IV, A, M) = \mathcal{D}_K(IV, A, C, T)$ if $\mathcal{V}_K(IV, A, C, T) = \top$, and $\perp$ otherwise.

The conversion in the other direction is not immediate. While the verification algorithm $\mathcal{V}$ can be easily "extracted" from $\overline{\mathcal{D}}$ (i.e., one can easily construct $\mathcal{V}$ using $\overline{\mathcal{D}}$ — just replace M with $\top$), it is not clear if one can always "naturally" extract the decryption algorithm $\mathcal{D}$ from $\overline{\mathcal{D}}$. However, all practical AE schemes that we are aware of can be constructed from a triplet $(\mathcal{E}, \mathcal{D}, \mathcal{V})$ as above, and hence their decryption algorithms $\overline{\mathcal{D}}$ are all naturally separable into $\mathcal{D}$ and $\mathcal{V}$.

3.2 Types of AE Schemes

Classification Based on IVs. In order to achieve semantic security [25], AE schemes must be probabilistic or stateful [5]. Usually the randomness or state is focused into an IV [36]. How the IV is used restricts the syntax of the scheme and the types of adversaries considered in the security notions:

1. **Random IV.** The environment chooses a random IV for each encryption, thus an adversary has no control over the choice of IV for each encryption. The generated IV must be sent along with the ciphertext so that the receiver can decrypt.
2. **Nonce IV.** A distinct IV is used for each encryption, thus an adversary can choose but does not repeat nonce IV values in its encryption queries. How the parties synchronize the nonce is left implicit.
3. **Arbitrary IV.** No restrictions on the IV are imposed, thus an adversary may choose any IV for encryption. Often a deterministic AE scheme does not even have an IV input, in which case an IV can be embedded into the associated data A, which gets authenticated along with the plaintext M but does not get encrypted; A is sent in the clear.

Table 2. The type of random oracle needed depending upon the class of AE scheme considered.

IV type	type of encryption	
	online	offline
random	random oracle	random oracle
nonce	random oracle	random oracle
arbitrary	random-up-to-prefix oracle	random-up-to-repetition oracle

In all IV cases the adversary can choose arbitrarily the IV input values to the decryption oracle. Towards formalizing the conventional AE definitions under the new syntax, we make implicit oracle distinctions depending on the IV type. Note that while for random or nonce IV schemes semantic security can be achieved, in the case of arbitrary IV, the AE scheme reduces to deterministic AE for repeated IVs. In the latter case, the two common notions are "*privacy up to repetition*" which is used for DAE [38] and "*privacy up to prefix*" which is used for authenticated online encryption [23]. In any case, we write \$ to indicate the ideal oracle from which an adversary tries to distinguish the real encryption oracle $\mathcal{E}_K$, where K is a secret key. Depending on the type of AE scheme that uses $\mathcal{E}$, the ideal \$ oracle should be either the random oracle, random-up-to-repetition oracle, or random-up-to-prefix oracle. Each of the cases with their respective random oracles are listed in Table 2. In order to avoid redundancy in the wording of the definitions, whenever we write $\Delta(\mathcal{E}_K, \ldots ; \$, \ldots)$, it is understood that the \$ oracle is the one appropriate for the AE scheme consisting of $\mathcal{E}$.

Online Encryption/Decryption Algorithms. A further distinction is made between schemes that are online versus schemes that are not online (as defined in Sect. 2). An AE scheme with online encryption is one in which the ciphertext can be output as the plaintext is received. Concretely, we require that for each (K, IV, A) the resulting encryption function is online as a function of the plaintext M.

Although decryption in AE schemes can never be online due to the fact that the message needs to be verified before it is output, we still consider schemes which can compute the plaintext as the ciphertext is received. In particular, a scheme with online decryption is one in which this plaintext-computing algorithm, viewed as a function of the ciphertext and tag input, is online. Note that in some schemes the tag could be received before the ciphertext, in which case we still consider $\mathcal{D}$ to be online (even though our new syntax implies that the tag is always received after the ciphertext).

3.3 Conventional Security Definitions under the New Syntax

Let $\Pi = (\mathcal{E}, \mathcal{D}, \mathcal{V})$ denote an AE scheme. With this new syntax we formulate the definitions of conventional security notions, IND-CPA, IND-CCA, and INT-CTXT. As mentioned above, the security notions will be defined in terms of an unspecified \$, where the exact nature of \$ depends on the type of IV allowed (cf. Table 2).

Definition 1 (IND-CPA Advantage). *Let* $\mathbf{D}$ *be a computationally bounded adversary with access to one oracle* $\mathcal{O}$*. Then the IND-CPA advantage of* $\mathbf{D}$ *relative to* Π *is given by*

$$\mathsf{CPA}_{\Pi}(\mathbf{D}) := \underset{\mathbf{D}}{\Delta}(\mathcal{E}_K \,;\, \$) \ ,$$

where $K \xleftarrow{R} \mathsf{K}$.

Recall from Sect. 2 that we mean that K is chosen uniformly at random from some set.

Definition 2 (IND-CCA Advantage). *Let* $\mathbf{D}$ *be a computationally bounded adversary with access to two oracles* $\mathcal{O}_1$ *and* $\mathcal{O}_2$*, such that* $\mathbf{D}$ *never queries* $\mathcal{O}_1 \hookrightarrow \mathcal{O}_2$ *nor* $\mathcal{O}_2 \hookrightarrow \mathcal{O}_1$*. Then the IND-CCA advantage of* $\mathbf{D}$ *relative to* Π *is given by*

$$\mathsf{CCA}_{\Pi}(\mathbf{D}) := \underset{\mathbf{D}}{\Delta}(\mathcal{E}_K, \mathcal{D}_K \,;\, \$, \mathcal{D}_K) \ ,$$

where $K \xleftarrow{R} \mathsf{K}$*.*

Note that IND-CCA as defined above does not apply to the random IV setting. When a random IV is used, the adversary is not prohibited from querying $\mathcal{O}_2 \hookrightarrow \mathcal{O}_1$. We introduce a version of IND-CCA below, which can be applied to all random, nonce and arbitrary IV settings.

Definition 3 (IND-CCA′ Advantage). *Let* $\mathbf{D}$ *be an adversary as in Def. 2, except* $\mathbf{D}$ *may now query* $\mathcal{O}_2 \hookrightarrow \mathcal{O}_1$*. Then the IND-CCA′ advantage of* $\mathbf{D}$ *relative to* Π *is given by*

$$\mathsf{CCA}'_{\Pi}(\mathbf{D}) := \underset{\mathbf{D}}{\Delta}(\mathcal{E}_K, \mathcal{D}_K \,;\, \$, \mathcal{D}_K) \ ,$$

where $K \xleftarrow{R} \mathsf{K}$*.*

Definition 4 (INT-CTXT Advantage). *Let* $\mathbf{F}$ *be a computationally bounded adversary with access to two oracles* $\mathcal{E}_K$ *and* $\mathcal{V}_K$*, such that* $\mathbf{F}$ *never queries* $\mathcal{E}_K \hookrightarrow \mathcal{V}_K$*. Then the INT-CTXT advantage of* $\mathbf{F}$ *relative to* Π *is given by*

$$\mathsf{CTXT}_{\Pi}(\mathbf{F}) := \Pr\left[\mathbf{F}^{\mathcal{E}_K, \mathcal{V}_K} \text{ forges}\right] \ ,$$

where the probability is defined over the random key K *and random coins of* $\mathbf{F}$*. Here,* "forges" *means the event of the oracle* $\mathcal{V}_K$ *returning* $\top$ *to the adversary.*

4 Security Under Release of Unverified Plaintext

4.1 Security of Encryption

We introduce the notion of plaintext-aware encryption of symmetric-key encryption schemes. An analysis of existing plaintext-aware schemes can be found in Sect. 6. The formalization is similar to the one in the public-key setting [10]. Let $\Pi = (\mathcal{E}, \mathcal{D})$ denote an encryption scheme.

Definition 5 (PA1 Advantage). *Let* $\mathbf{D}$ *be an adversary with access to two oracles* $\mathcal{O}_1$ *and* $\mathcal{O}_2$*. Let* $\mathbf{E}$ *be an algorithm with access to the history of queries made to* $\mathcal{O}_1$ *by* $\mathbf{D}$*, called a* PA1-extractor*. We allow* $\mathbf{E}$ *to maintain state across invocations. The PA1 advantage of* $\mathbf{D}$ *relative to* $\mathbf{E}$ *and* Π *is*

$$\mathsf{PA1}^{\mathbf{E}}_{\Pi}(\mathbf{D}) := \underset{\mathbf{D}}{\Delta}(\mathcal{E}_K, \mathcal{D}_K \,;\, \mathcal{E}_K, \mathbf{E}) \ ,$$

where $K \xleftarrow{R} \mathsf{K}$*, and the probability is defined over the key* K*, the random coins of* $\mathbf{D}$*, and the random coins of* $\mathbf{E}$*.*

The adversary $\mathbf{D}$ tries to distinguish the case in which its second oracle $\mathcal{O}_2$ is given by $\mathcal{D}_K$ versus the case in which $\mathcal{O}_2$ is given by $\mathbf{E}$. The task of $\mathbf{E}$ is to mimic the outputs of $\mathcal{D}_K$ given only the history of queries made to $\mathcal{E}_K$ by $\mathbf{D}$ (the key is not given to $\mathbf{E}$). Note that $\mathbf{D}$ is allowed to make queries of the form $\mathcal{E}_K \hookrightarrow \mathbf{E}$; these can easily be answered by $\mathbf{E}$ via the query history.

PA2 is a strengthening of PA1 where the extractor no longer has access to the query history of $\mathcal{E}_K$. Note that in order for this to work, we cannot allow the adversaries to make queries of the form $\mathcal{E}_K \hookrightarrow \mathbf{E}$.

Definition 6 (PA2 Advantage). *Let* $\mathbf{D}$ *be an adversary as in Def. 5, with the added restriction that it may not ask queries of the form* $\mathcal{O}_1 \hookrightarrow \mathcal{O}_2$*. Let* $\mathbf{E}$ *be an algorithm, called a* PA2-extractor*. We allow* $\mathbf{E}$ *to maintain state across invocations. The PA2 advantage of* $\mathbf{D}$ *relative to* $\mathbf{E}$ *and* Π *is*

$$\mathsf{PA2}^{\mathbf{E}}_{\Pi}(\mathbf{D}) := \underset{\mathbf{D}}{\Delta}(\mathcal{E}_K, \mathcal{D}_K \,;\, \mathcal{E}_K, \mathbf{E}) \ ,$$

where $K \xleftarrow{R} \mathsf{K}$*, and the probability is defined over the key* K*, the random coins of* $\mathbf{D}$*, and the random coins of* $\mathbf{E}$*.*

An equivalent way of describing PA2 is via *decryption independence* (DI), which means that the adversary cannot distinguish between encryption and decryption under the same key and under different keys.

Definition 7 (Decryption Independence). *Let* $\mathbf{D}$ *be a distinguisher accepting two oracles not making queries of the form* $\mathcal{O}_1 \hookrightarrow \mathcal{O}_2$*, then the DI advantage of* $\mathbf{D}$ *relative to* Π *is*

$$\mathsf{DI}_{\Pi}(\mathbf{D}) := \underset{\mathbf{D}}{\Delta}(\mathcal{E}_K, \mathcal{D}_K \,;\, \mathcal{E}_K, \mathcal{D}_L) \ ,$$

where $K, L \xleftarrow{R} \mathsf{K}$ *are independent.*

4.2 Security of Verification

Integrity when releasing unverified plaintext is a modification of INT-CTXT (Def. 4) to include the decryption oracle as a means to obtain unverified plaintext. Let $\Pi = (\mathcal{E}, \mathcal{D}, \mathcal{V})$ be an AE scheme with separate decryption and verification.

Definition 8 (INT-RUP Advantage). *Let* $\mathbf{F}$ *be a computationally bounded adversary with access to three oracles* $\mathcal{E}_K$*,* $\mathcal{D}_K$*, and* $\mathcal{V}_K$*, such that* $\mathbf{F}$ *never queries* $\mathcal{E}_K \hookrightarrow \mathcal{V}_K$*. Then the INT-RUP advantage of* $\mathbf{F}$ *relative to* Π *is given by*

$$\mathsf{INT\text{-}RUP}_{\Pi}(\mathbf{F}) := \Pr\left[\mathbf{F}^{\mathcal{E}_K, \mathcal{D}_K, \mathcal{V}_K} \text{ forges}\right] \ ,$$

where the probability is defined over the key K *and random coins of* $\mathbf{F}$*. Here,* "forges" *means the event of the oracle* $\mathcal{V}_K$ *returning* $\top$ *to the adversary.*

5 Relations Among Notions

In this section we study the relations among the plaintext awareness notions and the conventional security notions for encryption (see Sect. 3.3). The results are separated in the security of encryption (Sect. 5.1) and the security of verification (Sect. 5.2).

5.1 Security of Encryption

We start off with a theorem justifying the naming of the PA notions.

Theorem 1 (PA2 $\Rightarrow$ PA1). *Let* Π *be an encryption scheme that is PA2. Then* Π *is PA1.*

Proof. Let Π be an encryption scheme that is PA2 (the proof holds for any type of IV). Let $\mathbf{E}_2$ be the PA2-extractor associated to Π. We define the PA1-extractor $\mathbf{E}_1$ as follows. Let C be the input given to $\mathbf{E}_1$, then $\mathbf{E}_1$ checks to see if C is in the query history of $\mathcal{E}_K$.

1. If C is in the query history, then $\mathbf{E}_1$ returns the corresponding plaintext from the history.
2. If C is not in the query history, then $\mathbf{E}_1$ returns $\mathbf{E}_2(C)$.

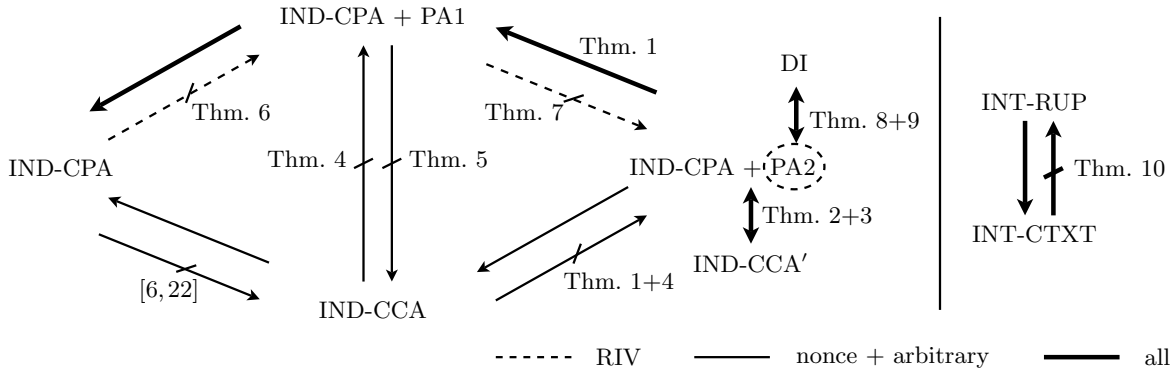

Fig. 2. Implications and separations between the IND-CPA, IND-CPA+PA1, IND-CPA+PA2, IND-CCA, IND-CCA′, PA2, and DI security notions (left) and INT-CTXT and INT-RUP (right). Dashed lines refer to relations that hold if the IV is random and thin solid lines in case of nonce or arbitrary IV. We use a thick solid line if the relation holds under all IV cases.

Let $\mathbf{D}_1$ be a PA1 adversary for $\mathbf{E}_1$. We construct a PA2 adversary $\mathbf{D}_2$ as follows: $\mathbf{D}_2$ runs $\mathbf{D}_1$ and directly forwards $\mathbf{D}_1$'s oracle queries to its own oracles. Exactly like $\mathbf{E}_1$, the adversary $\mathbf{D}_2$ responds to ciphertexts C which are in the query history of $\mathcal{E}_K$ with the corresponding plaintext from the history.

The PA1 game is perfectly simulated by $\mathbf{D}_2$, hence

$$\mathsf{PA1}^{\mathbf{E}_1}_{\Pi}(\mathbf{D}_1) \leq \mathsf{PA2}^{\mathbf{E}_2}_{\Pi}(\mathbf{D}_2) \ .$$

□

As in the public-key setting, PA2 along with IND-CPA implies IND-CCA′.

Theorem 2 (IND-CPA + PA2 ⇒ IND-CCA′). *Let Π be an encryption scheme that is PA2 secure and IND-CPA secure. Then Π is IND-CCA′ secure.*

Proof. Let $\mathbf{E}$ be the PA2-extractor given with Π (the proof holds for any type of IV). Let $\mathbf{D}$ be an adversary as in Def. 3. By the triangle inequality,

$$\begin{aligned}\mathsf{CCA}'_{\Pi}(\mathbf{D}) &= \underset{\mathbf{D}}{\Delta}(\mathcal{E}_K, \mathcal{D}_K \,;\, \$, \mathcal{D}_K) \ , \\ &\leq \underset{\mathbf{D}}{\Delta}(\mathcal{E}_K, \mathcal{D}_K \,;\, \mathcal{E}_K, \mathbf{E}) + \underset{\mathbf{D}}{\Delta}(\mathcal{E}_K, \mathbf{E} \,;\, \$, \mathbf{E}) + \underset{\mathbf{D}}{\Delta}(\$, \mathbf{E} \,;\, \$, \mathcal{D}_K) \ . \end{aligned} \tag{1}$$

Extractor $\mathbf{E}$ is independent of \$, hence

$$\underset{\mathbf{D}}{\Delta}(\$, \mathbf{E} \,;\, \$, \mathcal{D}_K) \leq \underset{\mathbf{D}_1}{\Delta}(\mathbf{E} \,;\, \mathcal{D}_K) \ ,$$

where $\mathbf{D}_1$ simulates $\mathbf{D}$'s \$-queries. Note that $\mathbf{D}_1$ can be viewed as a PA2-adversary, hence

$$\underset{\mathbf{D}_1}{\Delta}(\mathbf{E} \,;\, \mathcal{D}_K) \leq \mathsf{PA2}^{\mathbf{E}}_{\Pi}(\mathbf{D}_1) \ .$$

Furthermore, since $\mathbf{E}$ is independent of $\mathcal{E}_K$,

$$\underset{\mathbf{D}}{\Delta}(\mathcal{E}_K, \mathbf{E} \,;\, \$, \mathbf{E}) \leq \mathsf{CPA}_{\Pi}(\mathbf{D}_2) \ ,$$

for some $\mathbf{D}_2$ which simulates $\mathbf{E}$. Since the first term in (1) is just $\mathsf{PA2}^{\mathbf{E}}_{\Pi}(\mathbf{D})$, we get

$$\mathsf{CCA}'_{\Pi}(\mathbf{D}) \leq \mathsf{PA2}^{\mathbf{E}}_{\Pi}(\mathbf{D}) + \mathsf{PA2}^{\mathbf{E}}_{\Pi}(\mathbf{D}_1) + \mathsf{CPA}_{\Pi}(\mathbf{D}_2) \ .$$

□

Yet the similarities between public-key and symmetric-key stop there. We have that PA2 + IND-CPA is in fact equivalent with IND-CCA′.

Theorem 3 (IND-CCA′ ⇒ PA2). *Let Π be an encryption scheme that is IND-CCA′ secure. Then Π is PA2 secure.*

Proof. Let Π be an encryption scheme that is IND-CCA′ (with any type of IV). Let $\mathbf{E} := \mathcal{D}_{K'}$ for some random key K'. Let $\mathbf{D}$ be an adversary as in Def. 6. By the triangle inequality,

$$\begin{aligned}\mathsf{PA2}^{\mathbf{E}}_{\Pi}(\mathbf{D}) &= \underset{\mathbf{D}}{\Delta}(\mathcal{E}_K, \mathcal{D}_K \,;\, \mathcal{E}_K, \mathcal{D}_{K'}) \ , \\ &\leq \underset{\mathbf{D}}{\Delta}(\mathcal{E}_K, \mathcal{D}_K \,;\, \$, \mathcal{D}_K) + \underset{\mathbf{D}}{\Delta}(\$, \mathcal{D}_K \,;\, \mathcal{E}_K, \mathcal{D}_{K'}) \ .\end{aligned}$$

The first term is $\mathsf{CCA}'_{\Pi}(\mathbf{D})$. Furthermore, note that

$$\underset{\mathbf{D}}{\Delta}(\$, \mathcal{D}_K \,;\, \mathcal{E}_K, \mathcal{D}_{K'}) \leq \mathsf{CPA}_{\Pi}(\mathbf{D}') \ ,$$

for some adversary $\mathbf{D}'$, as $\mathbf{D}'$ can just simulate $\mathcal{D}_{K'}$. Therefore:

$$\mathsf{PA2}^{\mathbf{E}}_{\Pi}(\mathbf{D}) \leq \mathsf{CCA}'_{\Pi}(\mathbf{D}) + \mathsf{CPA}_{\Pi}(\mathbf{D}_1) \ .$$

□

Note that the above theorems all hold under each IV situation (random, nonce, arbitrary). The relation between plaintext awareness and IND-CCA only makes sense for the nonce and arbitrary IV schemes due to the fact that IND-CCA security is not defined for random IV. Here we actually have a separation both ways:

Theorem 4 (IND-CCA ⇏ IND-CPA + PA1). *Assume there exists a nonce or arbitrary IV IND-CCA-secure encryption scheme. Then there exists an IND-CCA-secure encryption scheme that is* not *PA1 secure.*

Proof. Let $\Pi = (\mathcal{E}, \mathcal{D})$ denote a nonce or arbitrary IV symmetric encryption scheme that is IND-CCA.

Let N_0 be some fixed IV and C_0 some fixed ciphertext with $\mathcal{D}_K(N_0, C_0) = M_K$. Define $\widetilde{\Pi} = (\widetilde{\mathcal{E}}, \widetilde{\mathcal{D}})$ as

$$\begin{aligned}\widetilde{\mathcal{E}}_K(N, M) &= \begin{cases} C_0 & \text{if } M = M_K \text{ and } N = N_0 \ , \\ \mathcal{E}_K(N, M) & \text{otherwise} \ , \end{cases} \\ \widetilde{\mathcal{D}}_K(N, C) &= \mathcal{D}_K(N, C) \ .\end{aligned}$$

Claim. There exists an adversary $\mathbf{D}$ such that for every extractor $\mathbf{E}$, $\mathbf{D}$ succeeds in breaking the PA1 security of $\widetilde{\Pi}$ with a high probability.

The adversary $\mathbf{D}$ simply queries $\mathcal{O}_1(N_0, \mathcal{O}_2(N_0, C_0))$:

1. If $\mathcal{O}_2 = \widetilde{\mathcal{D}}_K$, $\mathbf{D}$ queries $\widetilde{\mathcal{E}}_K(N_0, \widetilde{\mathcal{D}}_K(N_0, C_0))$, which always equals C_0.
2. If $\mathcal{O}_2 = \mathbf{E}$, $\mathbf{D}$ queries $\widetilde{\mathcal{E}}_K(N_0, \mathbf{E}(N_0, C_0))$. This will only equal C_0 if $\mathbf{E}$ finds M_K such that $\mathcal{E}_K(N_0, M_K) = C_0$. This probability can be upper bounded by the CPA advantage of Π via an adversary $\mathbf{A}$ which checks to see if $\mathcal{O}(N_0, \mathbf{E}(N_0, C_0)) = C_0$ (if the equality holds $\mathbf{A}$ guesses that its oracle is $\mathcal{E}_K$).

Claim. Let $\mathbf{A}$ be an IND-CCA adversary of $\widetilde{\Pi}$. Then there exist IND-CCA adversaries $\mathbf{B}$ and $\mathbf{C}$ of Π with the same or smaller running time than $\mathbf{A}$, such that

$$\mathsf{CCA}_{\widetilde{\Pi}}(\mathbf{A}) \leq \mathsf{CCA}_{\Pi}(\mathbf{B}) + \mathsf{CCA}_{\Pi}(\mathbf{C}) \ .$$

We construct $\mathbf{B}$ as follows: $\mathbf{B}$ runs $\mathbf{A}$ and answers $\mathbf{A}$'s $\widetilde{\mathcal{E}}_K$ query with $\mathcal{E}_K$ and $\mathbf{A}$'s $\widetilde{\mathcal{D}}_K$ queries with $\mathcal{D}_K$ (or with the corresponding \$ oracle). Note that $\mathbf{B}$'s simulation of $\mathbf{A}$'s IND-CCA game is perfect unless $\mathbf{A}$ induces $\mathbf{B}$ to make the query $\mathcal{E}_K(N_0, \mathcal{D}_K(N_0, C_0))$. Since $\mathbf{A}$ cannot ask queries of the form $\widetilde{\mathcal{E}}_K(N, \widetilde{\mathcal{D}}_K(N, C))$, $\mathbf{A}$ must find (N_1, C_1) such that

$$\mathcal{D}_K(N_1, C_1) = \mathcal{D}_K(N_0, C_0) \ ,$$

which in turn allows us to create another IND-CCA adversary $\mathbf{C}$ for Π. □

Theorem 5 (IND-CPA + PA1 $\not\Rightarrow$ IND-CCA). *Assume there exists a nonce or arbitrary IV PA1 secure and IND-CPA secure encryption scheme. Then there exists a PA1 secure and IND-CPA secure encryption scheme that is* not *IND-CCA secure.*

Proof. Let $\Pi = (\mathcal{E}, \mathcal{D})$ be a nonce IV encryption scheme that is IND-CPA and PA1 secure with extractor $\mathbf{E}$. Define $b_K(N, M)$ to be the first bit of $\mathcal{E}_K(N, \mathcal{E}_K(N, M))$. Let n be an integer. Let $m := \max_{K,N,M \in \{0,1\}^n} |\mathcal{E}_K(N, M)|$ and let $\mathrm{len}(k)$ denote a $\log_2 m$-bit representation of k. Let $\widetilde{\Pi}$ be defined as follows:

$$\begin{aligned} \widetilde{\mathcal{E}}_K(N, M) &= b_K(N, M) \parallel \mathcal{E}_K(N, \mathrm{len}(|\mathcal{E}_K(N, M)|) \| \mathcal{E}_K(N, M) \| M) \ , \\ \widetilde{\mathcal{D}}_K(N, b \parallel C) &= M' \ , \end{aligned}$$

where M' is $\mathcal{D}_K(N, C)$ with $\mathrm{len}(|C|)\|C$ removed. Then $\widetilde{\Pi}$ is IND-CPA and PA1 with extractor $\widetilde{\mathbf{E}}(N, b \parallel C)$ outputting M' where M' is $\mathbf{E}(N, C)$ with $\mathrm{len}(|C|)\|C$ removed.

Furthermore, $\widetilde{\Pi}$ is not IND-CCA because an IND-CCA adversary can query $\mathcal{O}_1(N, M) = b \parallel C$, and then query $\widetilde{\mathcal{D}}_K(N, \overline{b} \parallel C)$ to get a distinguishing event (here $\overline{b}$ is the complement of b).

□

Finally, although we can deduce separations between IND-CPA and PA1 via the above theorems for the nonce and arbitrary IV cases, we need to separately prove the separations for the random IV case.

Theorem 6 (IND-CPA $\not\Rightarrow$ PA1). *Assume there exists a random IV IND-CPA-secure encryption scheme. Then there exists an IND-CPA-secure encryption scheme that is* not *PA1 secure.*

Proof. Let $\Pi = (\mathcal{E}, \mathcal{D})$ denote a random IV encryption scheme that is IND-CPA. Let H_0 be some constant from the space of IVs of Π. Say that $\mathcal{E}_K(H_0, K) = (H_0, C_K)$ is the encryption of K when the IV is H_0, and that $\mathcal{D}_K(H_0, C_0) = M_K$ is the decryption of some constant C_0 under K and H_0. Define $\widetilde{\Pi} = (\widetilde{\mathcal{E}}, \widetilde{\mathcal{D}})$ as

$$\widetilde{\mathcal{E}}_K(H, M) = \begin{cases} (H_0, C_K) & \text{if } H = H_0 \text{ and } M = M_K \ , \\ (H_0, C_0) & \text{if } H = H_0 \text{ and } M = K \ , \\ \mathcal{E}_K(H, M) & \text{otherwise} \ , \end{cases}$$

$$\widetilde{\mathcal{D}}_K(H, C) = \begin{cases} K & \text{if } H = H_0 \text{ and } C = C_0 \ , \\ M_K & \text{if } H = H_0 \text{ and } C = C_K \ , \\ \mathcal{D}_K(H, C) & \text{otherwise} \ . \end{cases}$$

Claim. There exists an adversary $\mathbf{D}$ such that for every extractor $\mathbf{E}$, $\mathsf{PA1}^{\mathbf{E}}_{\widetilde{\Pi}}(\mathbf{D})$ is non-negligible.

The adversary queries $\mathcal{O}_2(H_0, C_0)$:

1. If $\mathcal{O}_2 = \widetilde{\mathcal{D}}_K$, then $\widetilde{\mathcal{D}}_K(H_0, C_0)$ always equals K.

2. If $\mathcal{O}_2 = \mathbf{E}$, then $\mathbf{E}$ must somehow output K. The probability of $\mathbf{E}$ doing this is upper bounded by the CPA security of Π.

Claim. Let $\mathbf{A}$ be an IND-CPA adversary for $\widetilde{\Pi}$. Then there exists an IND-CPA adversary $\mathbf{B}$ for Π such that

$$\mathsf{CPA}_{\widetilde{\Pi}}(\mathbf{A}) \leq \mathsf{CPA}_{\Pi}(\mathbf{B}) + \frac{q}{m} \ ,$$

where m is the size of the set of IVs from which the random IV is chosen and q is the number of queries that $\mathbf{A}$ makes to the encryption oracle.

Adversary $\mathbf{B}$ forwards $\mathbf{A}$'s oracle queries to its own oracles. The simulation of IND-CPA for $\mathbf{A}$ is perfect unless $\mathbf{A}$ queries K or M_K under H_0. The probability of H_0 being used as an IV is $1/m$. □

Theorem 7 (IND-CPA + PA1 $\not\Rightarrow$ PA2). *Assume there exists a random IV PA1 secure and IND-CPA secure encryption scheme. Then there exists PA1 secure and IND-CPA secure encryption scheme that is not PA2 secure.*

The proof is similar to the one of Thm. 5.

Then we also have the equivalence between PA2 and DI.

Theorem 8 (PA2 $\Rightarrow$ DI). *Let Π be an encryption scheme that is PA2. Then Π is DI.*

Proof. Let $\mathbf{E}$ be the PA2-extractor given with Π (the proof holds for any type of IV). Let $\mathbf{D}$ be an adversary as in Def. 7. By the triangle inequality,

$$\begin{aligned}\mathsf{DI}_{\Pi}(\mathbf{D}) &= \underset{\mathbf{D}}{\triangle}(\mathcal{E}_K, \mathcal{D}_K \,;\, \mathcal{E}_K, \mathcal{D}_L) \ , \\ &\leq \underset{\mathbf{D}}{\triangle}(\mathcal{E}_K, \mathcal{D}_K \,;\, \mathcal{E}_K, \mathbf{E}) + \underset{\mathbf{D}}{\triangle}(\mathcal{E}_K, \mathbf{E} \,;\, \mathcal{E}_K, \mathcal{D}_L) \ .\end{aligned}$$

The first term is $\mathsf{PA2}^{\mathbf{E}}_{\Pi}(\mathbf{D})$. Furthermore, note that extractor $\mathbf{E}$ and $\mathcal{D}_L$ are independent of $\mathcal{E}_K$, hence

$$\underset{\mathbf{D}}{\triangle}(\mathcal{E}_K, \mathbf{E} \,;\, \mathcal{E}_K, \mathcal{D}_L) \leq \underset{\mathbf{D}_1}{\triangle}(\mathbf{E} \,;\, \mathcal{D}_L) \ ,$$

where $\mathbf{D}_1$ simulates $\mathbf{D}$'s $\mathcal{E}_K$-queries using some random key K. Note that $\mathbf{D}_1$ can be viewed as a PA2-adversary, hence

$$\underset{\mathbf{D}_1}{\triangle}(\mathbf{E} \,;\, \mathcal{D}_L) \leq \mathsf{PA2}^{\mathbf{E}}_{\Pi}(\mathbf{D}_1) \ .$$

Therefore

$$\mathsf{DI}_{\Pi}(\mathbf{D}) \leq \mathsf{PA2}^{\mathbf{E}}_{\Pi}(\mathbf{D}) + \mathsf{PA2}^{\mathbf{E}}_{\Pi}(\mathbf{D}_1) \ .$$

□

Theorem 9 (DI $\Rightarrow$ PA2). *Let Π be an encryption scheme that is DI. Then Π is PA2.*

Proof. Let Π be an encryption scheme that is DI (the proof holds for any type of IV). Let $\mathbf{E} := \mathcal{D}_L$ for some random key L. Let $\mathbf{D}$ be an adversary as in Def. 6.

$$\mathsf{PA2}^{\mathbf{E}}_{\Pi}(\mathbf{D}) = \underset{\mathbf{D}}{\triangle}(\mathcal{E}_K, \mathcal{D}_K \,;\, \mathcal{E}_K, \mathcal{D}_L) = \mathsf{DI}_{\Pi}(\mathbf{D}) \ .$$

□

5.2 Security of Verification

INT-RUP clearly implies INT-CTXT. The opposite is, however, not necessarily true.

Theorem 10 (INT-CTXT $\not\Rightarrow$ INT-RUP). *Assume there exists an INT-CTXT secure encryption scheme. Then there exists an INT-CTXT secure encryption scheme that is not INT-RUP secure.*

Proof. Let $\Pi = (\mathcal{E}, \mathcal{D}, \mathcal{V})$ be an authenticated encryption scheme that is INT-CTXT secure (the proof holds for any type of IV). Let (N_0, M_0) be some IV and plaintext pair, and let (N_1, C_1, T_1) and (N_2, C_2, T_2) be some IV, ciphertext, and tag tuples. Let $\mathcal{E}_K(N_0, M_0) = (N_0, C_0, T_0)$, and let $\widetilde{\Pi} = (\widetilde{\mathcal{E}}, \widetilde{\mathcal{D}}, \widetilde{\mathcal{V}})$ be defined as follows:

$$\widetilde{\mathcal{E}}_K(N, M) = \mathcal{E}_K(N, M)$$

$$\widetilde{\mathcal{D}}_K(N, C, T) = \begin{cases} (N_1, C_0) & \text{if } (N, C, T) = (N_1, C_1, T_1) \ , \\ (N_2, T_0) & \text{if } (N, C, T) = (N_2, C_2, T_2) \ , \\ \mathcal{D}_K(N, C, T) & \text{otherwise} \ , \end{cases}$$

$$\widetilde{\mathcal{V}}_K(N, C, T) = \mathcal{V}_K(N, C, T) \ .$$

The correctness condition is still satisfied by $\widetilde{\Pi}$. Clearly, $\widetilde{\Pi}$ is INT-CTXT. However, it is not INT-RUP secure: a forger queries $\widetilde{\mathcal{D}}_K(N_1, C_1, T_1) = (N_1, C_0)$ and $\widetilde{\mathcal{D}}_K(N_2, C_2, T_2) = (N_2, T_0)$ and uses these tuples to query $\mathcal{V}_K(N_0, C_0, T_0)$ so as to forge $\widetilde{\Pi}$. □

6 Achieving Plaintext Awareness

6.1 Why Existing Schemes Do Not Achieve PA1

In conventional AE schemes such as OCB, GCM, SpongeWrap, CCM, COPA, and McOE-G, a ciphertext is computed using some bijective function, and then a tag is appended to the ciphertext. The schemes achieve AE because the tag prevents all ciphertexts from being valid. But if the tag is no longer checked, then we cannot achieve PA1, as explained below.

Let $\Pi = (\mathcal{E}_K, \mathcal{D}_K)$ be a nonce or arbitrary IV encryption scheme, then we can describe Π as follows,

$$\mathcal{E}_K(IV, A, M) = E_K(IV, A, M) \parallel F_K(IV, A, M) \ ,$$

where E_K is length-preserving, i.e. $|E_K(IV, A, M)| = |M|$. One can view $F_K(IV, A, M)$ as the tag-producing function from a scheme such as GCM. In the following proposition we prove that if Π is IND-CPA and PA1, then E_K cannot be bijective for each (IV, A), assuming either a nonce or arbitrary IV. Note that the proposition only holds if Π is a nonce or arbitrary IV scheme.

Proposition 1. *Say that E_K is bijective for all (IV, A), then there exists an adversary $\mathbf{D}$ such that for all extractors $\mathbf{E}$, there exists an adversary $\mathbf{D}_1$ such that*

$$1 - \mathsf{CPA}_{\Pi}(\mathbf{D}_1) \leq \mathsf{PA1}^{\mathbf{E}}_{\Pi}(\mathbf{D}) \ ,$$

where $\mathbf{D}$ makes one $\mathcal{O}_1$ query, one $\mathcal{O}_2$ query, and $\mathbf{D}_1$ is as efficient as $\mathbf{D}$ plus one query to $\mathbf{E}$.

Proof. Since E_K is bijective, we know that for all (K, IV, A, C),

$$E_K(IV, A, \mathcal{D}_K(IV, A, C)) = C' \ ,$$

where C' is a prefix for C of length $|\mathcal{D}_K(IV, A, C)|$. Define $\mathbf{D}$ as follows. It generates C uniformly at random from $\{0,1\}^n$ and queries $\mathcal{O}_2(IV, A, C)$, where IV and A are arbitrary constants,

receiving M as output. Then $\mathbf{D}$ queries $\mathcal{O}_1(IV, A, M)$ to receive C'. The adversary $\mathbf{D}$ outputs 1 if the first $|M|$ bits of C' equals C, and 0 otherwise.

Let $\mathbf{E}$ be an extractor for Π. The probability that $\mathbf{D}$ outputs 1 when $\mathcal{O}_2 = \mathbf{E}$ is equal to the probability that $\mathbf{E}$ can output a message M such that $E_K(IV, A, M) = C$ without $\mathcal{E}$ ever having been queried. This in turn is upper bounded by the probability that $\mathbf{D}_1$ wins the CPA_Π game, where $\mathbf{D}_1$ is an IND-CPA adversary which runs $\mathbf{D}$, answers $\mathbf{D}$'s $\mathcal{O}_2$ queries using $\mathbf{E}$, and outputs 1 if $\mathbf{E}$ outputs the correct message M. □

We conclude that in order for a nonce or arbitrary IV scheme to be PA1 and IND-CPA, E_K must either not be bijective, or not be length-preserving.

6.2 PA1 Random IV Schemes

We illustrate Def. 5 and the idea of an extractor by considering the CTR mode with a random IV. An extractor for CBC mode is described in App. A.

Example 1 (RIV-CTR Extractor). Let $F : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ be a PRF. For $M_i \in \{0,1\}^n$, $1 \le i \le \ell$, define RIV-CTR encryption as

$$\mathcal{E}_K(M_1 \cdots M_\ell) = F_K(C_0 + 1) \oplus M_1 \parallel \cdots \parallel F_K(C_0 + \ell) \oplus M_\ell \ ,$$

where C_0 is selected uniformly at random from $\{0,1\}^n$ for each encryption, and decryption as

$$\mathcal{D}_K(C_0, C_1 \cdots C_\ell) = F_K(C_0 + 1) \oplus C_1 \parallel \cdots \parallel F_K(C_0 + \ell) \oplus C_\ell \ .$$

Note that we have suppressed the explicit IV input to $\mathcal{E}_K$ and use C_0 instead.

We can define an extractor $\mathbf{E}$ for RIV-CTR as follows. Initially, $\mathbf{E}$ generates a random key K' which it will use via $F_{K'}$. Let $(C_0, C_1 \cdots C_\ell)$ denote an input to $\mathbf{E}$. Using C_0, the extractor searches its history for a ciphertext with C_0 as IV.

1. If such a ciphertext exists, we let $(C'_1 \cdots C'_m, M'_1 \cdots M'_m)$ denote the longest corresponding $\mathcal{E}_K$ query-response pair. Define $\kappa_i := C'_i \oplus M'_i$ for $1 \le i \le \min\{\ell, m\}$. Notice that κ_i corresponds to the keystream generated by F_K for $1 \le i \le \ell$. For $m < i \le \ell$ we generate κ_i by $F_{K'}(C_0 + i)$.
2. If there is no such ciphertext, then we generate κ_i as $F_{K'}(C_0 + i)$ for $1 \le i \le \ell$.

Then we set

$$\mathbf{E}(C_0, C_1 \cdots C_\ell) = C_1 \oplus \kappa_1, C_2 \oplus \kappa_2 \parallel \cdots \parallel C_\ell \oplus \kappa_\ell \ .$$

Proposition 2. *Let $\mathbf{D}$ be a PA1 adversary for RIV-CTR or RIV-CBC making queries whose lengths in number of blocks sum up to σ, then*

$$\mathsf{PA1}^{\mathbf{E}}_{\textit{RIV-CTR}}(\mathbf{D}) \le \underset{\mathbf{D}_1}{\Delta}(F_K, F_K\,;\, F_K, F_{K'}) + \frac{\sigma^2}{2^n} \ ,$$

where $\mathbf{D}_1$ is an adversary which may not make the same query to both of its oracles, and makes a total of σ queries with the same running time as $\mathbf{D}$.

The proof of this proposition can be found in App. A, along with CBC mode.

From Thm. 6 we know that IND-CPA does not imply PA1, but the example used in the proof is pathological since the decryption algorithm leaks the key when queried with the appropriate query. We do not know of a non-pathological example of a random IV scheme which does not achieve PA1.

In the following subsections we discuss ways of achieving PA1 assuming a nonce and arbitrary IV. Our basic building block will be a random IV PA1 scheme.

6.3 PA1 Nonce IV Schemes

Nonce IV schemes are not necessarily PA1 in general. For example, CTR mode with a nonce IV is not PA1 and Thm. 6 shows that IND-CPA is distinct from PA1. Furthermore, coming up with a generic technique which transforms nonce IV schemes into PA1 schemes in an efficient manner is most likely not possible.

If we assume that the nonce IV scheme, when used as a random IV scheme, is PA1, then there is an efficient way of making the nonce IV scheme PA1. Note that we already have an example of a scheme satisfying our assumption: nonce IV CTR mode is not PA1, but RIV-CTR is.

Nonce Decoy. The *nonce decoy* method creates a random-looking IV from the nonce IV and forces the decryption algorithm to use the newly generated IV. Note that we are not only transforming the nonce into a random nonce: the solution depends entirely on the fact that the decryption algorithm does *not* recompute the newly generated IV from the nonce IV.

Let $\Pi = (\mathcal{E}, \mathcal{D}, \mathcal{V})$ be a nonce-IV-based AE scheme. For simplicity assume $\mathsf{IV} := \{0,1\}^n$, so that IVs are of a fixed length n. We prepare a pseudo-random function $G_{K'} : \mathsf{IV} \to \mathsf{IV}$ with an independent key K'. We then construct an AE scheme $\Pi^* = (\mathcal{E}^*, \mathcal{D}^*, \mathcal{V}^*)$ as follows.

$\underline{\mathcal{E}^*_{K,K'}(IV, A, M)}$:
$IV^* \leftarrow G_{K'}(IV)$
$(C, T) \leftarrow \mathcal{E}_K(IV^*, A, M)$
$C^* \leftarrow IV^* \| C$
return (C^*, T)

$\underline{\mathcal{D}^*_{K,K'}(IV, A, C^*, T)}$:
$IV^* \| C \leftarrow C^*$
$M \leftarrow \mathcal{D}_K(IV^*, A, C, T)$
return M

$\underline{\mathcal{V}^*_{K,K'}(IV, A, C^*, T)}$:
$\widetilde{IV}^* \leftarrow G_{K'}(IV)$
$IV^* \| C \leftarrow C^*$
$b \leftarrow \mathcal{V}_K(IV^*, A, C, T)$
return $(\widetilde{IV}^* = IV^* \text{ and } b = \top)?\top : \bot$

Note that the decryption algorithm $\mathcal{D}^*$ does not make use of K' or IV. If the decryption algorithm recomputes IV^* using K' and IV, then Π^* will not be PA1. Furthermore, one can combine $\mathcal{D}^*$ and $\mathcal{V}^*$ in order to create a scheme which rejects ciphertexts when the IV it receives does not come from an encryption query.

First we show that Π with random IVs must be PA1 in order for Π^* to be PA1 (assuming G is a PRF).

Proposition 3 (If Π^* is PA1, then Π with random IVs is PA1). *Let $\mathbf{E}^*$ be a PA1-extractor for Π^* with nonce IV. Then there exists an extractor $\mathbf{E}$ for Π with random IV such that for all adversaries $\mathbf{D}$ there exist $\mathbf{D}_1$ and $\mathbf{D}_2$ such that*

$$\mathsf{PA1}^{\mathbf{E}}_{\Pi}(\mathbf{D}) \leq \mathsf{PA1}^{\mathbf{E}^*}_{\Pi^*}(\mathbf{D}_1) + \mathsf{PRF}_G(\mathbf{D}_2),$$

where $\mathbf{D}_1$ and $\mathbf{D}_2$ are as efficient as $\mathbf{D}$, and $\mathbf{E}$ is as efficient as $\mathbf{E}^*$.

Proof. Define $\mathbf{E}$ as follows. First $\mathbf{E}$ transforms its query history so that a query $\mathcal{E}_K(A, M) = (IV, A, C, T)$ is turned into $\mathcal{E}^*_{K,K'}(i, A, M) = (i, A, IV \| C, T)$, where i is a counter which is incremented for each encryption query. Then on input (IV, A, C, T), $\mathbf{E}$ responds with $\mathbf{E}^*(IV, A, IV \| C, T)$, where $\mathbf{E}^*$ is given $\mathbf{E}$'s transformed query history.

We denote $\mathbf{D}$'s oracles by $\mathcal{O}_1$ and $\mathcal{O}_2$, and $\mathbf{D}_1$'s oracles by $\mathcal{O}^*_1$ and $\mathcal{O}^*_2$. The adversary $\mathbf{D}_1$ runs $\mathbf{D}$ and maintains a counter i which is incremented for each query that $\mathbf{D}$ makes to $\mathcal{O}_1$. The oracle queries $\mathcal{O}_1(A, M)$ made by $\mathbf{D}$ are answered with $(C, T) := \mathcal{O}^*_1(i, A, M)$ where the first n bits are removed from C (so as to remove the IV^* value). The decryption oracle queries $\mathcal{O}_2(IV, A, C, T)$ are answered with $\mathcal{O}^*_2(IV, A, IV \| C, T)$.

If $\mathbf{D}$ can distinguish $G_{K'}(i)$ from uniformly distributed random bits, then we can construct an adversary $\mathbf{D}_2$ attacking the PRF advantage of G. Otherwise $\mathbf{D}_1$ simulates the PA1 game for $\mathbf{D}$ perfectly. □

Finally we show that Π being PA1 is sufficient in order to prove that Π^* is PA1 (again, assuming G is a PRF).

Proposition 4 (If Π with random IVs is PA1, then Π^* with nonce IV is PA1). *Let $\mathbf{E}$ be a PA1-extractor for Π with random IV. Then there exists an extractor $\mathbf{E}^*$ for Π^* with nonce IV such that for all adversaries $\mathbf{D}$ there exist $\mathbf{D}_1$ and $\mathbf{D}_2$ such that*

$$\mathsf{PA1}^{\mathbf{E}^*}_{\Pi^*}(\mathbf{D}) \leq \mathsf{PA1}^{\mathbf{E}}_{\Pi}(\mathbf{D}_1) + \mathsf{PRF}_G(\mathbf{D}_2),$$

where $\mathbf{D}_1$ and $\mathbf{D}_2$ are as efficient as $\mathbf{D}$, and $\mathbf{E}^*$ is as efficient as $\mathbf{E}$.

Proof. Define $\mathbf{E}^*$ as follows. First $\mathbf{E}^*$ transforms its history so that a query $\mathcal{E}^*_{K,K'}(IV, A, M) = (C^*, T)$ gets turned into $\mathcal{E}_K(A, M) = (IV^*, A, C, T)$, where $C^* = IV^* \| C$ and $|IV^*| = n$. Then on input (IV, A, C, T), $\mathbf{E}^*$ responds with $\mathbf{E}(IV^*, A, C', T)$, where $C = IV^* C'$ and $|IV^*| = n$, and $\mathbf{E}$ is run on the transformed query history.

Let $\mathcal{O}^*_1$ and $\mathcal{O}^*_2$ denote the oracles given to $\mathbf{D}$, and $\mathcal{O}_1$ and $\mathcal{O}_2$ the oracles given to $\mathbf{D}_1$. The adversary $\mathbf{D}_1$ runs $\mathbf{D}$ and answers a query $\mathcal{O}^*_1(IV, A, M)$ with $(IV \| C, T)$, where $(IV, C, T) = \mathcal{O}_1(A, M)$. A query $\mathcal{O}^*_2(IV, A, C, T)$ is answered with $\mathcal{O}_2(IV^*, A, C', T)$ where $C = IV^* C'$ and $|IV^*| = n$.

Again, $\mathbf{D}_1$ perfectly simulates the PA1 game for $\mathbf{D}$ as long as G is a PRF. □

In Sect. 7.2 we discuss what the nonce decoy does for INT-RUP.

6.4 PA1 Arbitrary IV Schemes.

PRF-to-IV. Using a technique similar to MAC-then-Encrypt [9], we can turn a random IV PA1 scheme into an arbitrary IV PA1 scheme.

The idea behind the *PRF-to-IV* method is to evaluate a VIL PRF over the input to the scheme and then to use the resulting output as an IV for the random IV encryption scheme. Let $\Pi = (\mathcal{E}, \mathcal{D}, \mathcal{V})$ be a random IV PA1 scheme taking IVs from $\{0,1\}^n$, and let $G : \{0,1\}^k \times \{0,1\}^* \to \{0,1\}^n$ be a VIL PRF.

$\underline{\mathcal{E}^*_{K,K'}(IV, A, M)}$:
$IV^* \leftarrow G_{K'}(IV \| A \| M)$
$(C, T) \leftarrow \mathcal{E}_K(IV^*, A, M)$
return $(C, IV^* \| T)$

$\underline{\mathcal{D}^*_{K,K'}(IV, A, C, IV^* \| T)}$:
$M \leftarrow \mathcal{D}_K(IV^*, A, C, T)$
return M

$\underline{\mathcal{V}^*_{K,K'}(IV, A, C, IV^* \| T)}$:
$M \leftarrow \mathcal{D}^*_{K,K'}(IV, A, C, IV^* \| T)$
$\widetilde{IV} \leftarrow G_{K'}(IV \| A \| M)$
return $\widetilde{IV} = IV^* \ ? \top : \bot$

The PRF-to-IV method is more robust than the nonce decoy since $\mathcal{D}^*$ really only can use IV^* to decrypt properly.

First we show that Π with random IV must be PA1 in order for Π^* to be PA1.

Proposition 5 (If Π^* is PA1, then Π with random IVs is PA1). *Let $\mathbf{E}^*$ be a PA1-extractor for Π^* with arbitrary IV. Then there exists an extractor $\mathbf{E}$ for Π with random IV such that for all adversaries $\mathbf{D}$ there exist $\mathbf{D}_1$ and $\mathbf{D}_2$ such that*

$$\mathsf{PA1}^{\mathbf{E}}_{\Pi}(\mathbf{D}) \leq \mathsf{PA1}^{\mathbf{E}^*}_{\Pi^*}(\mathbf{D}_1) + \mathsf{PRF}_G(\mathbf{D}_2),$$

where $\mathbf{D}_1$ and $\mathbf{D}_2$ are as efficient as $\mathbf{D}$, and $\mathbf{E}$ is as efficient as $\mathbf{E}^*$.

Proof. We define $\mathbf{E}$ as follows. On input (IV, A, C, T), $\mathbf{E}$ first transforms its query history by mapping $\mathcal{E}_K(A, M) = (IV, A, C, T)$ to $\mathcal{E}^*_K(i, A, M) = (i, A, C, IV \| T)$, where i is a counter which is incremented for each encryption query. Then it passes the transformed query history to $\mathbf{E}^*$. Finally $\mathbf{E}$ returns $\mathbf{E}^*(IV, A, C, IV \| T)$.

Let $\mathcal{O}_1$ and $\mathcal{O}_2$ denote the oracles given to $\mathbf{D}$, and $\mathcal{O}_1^*$ and $\mathcal{O}_2^*$ the oracles given to $\mathbf{D}_1$. The adversary $\mathbf{D}_1$ runs $\mathbf{D}$. On a query $\mathcal{O}_1(A, M)$ made by $\mathbf{D}$, $\mathbf{D}_1$ responds with (C, T), where $(C, IV^* \| T) = \mathcal{O}_1(i, A, M)$, where i is a counter which $\mathbf{D}_1$ maintains and increments for each query to $\mathcal{O}_1$. On a query $\mathcal{O}_2(IV, A, C, T)$ made by $\mathbf{D}$, $\mathbf{D}_1$ responds with $\mathcal{O}_2(IV, A, C, IV \| T^*)$.

Note that unless $\mathbf{D}$ distinguishes G from a uniform random function, $\mathbf{D}_1$ perfectly simulates the PRF game for $\mathbf{D}$. □

Finally, we show that Π being PA1 is sufficient as well.

Proposition 6 (If Π is PA1, then Π^* with arbitrary IVs is PA1). *Let $\mathbf{E}$ be a PA1-extractor for Π with random IV. Then there exists an extractor $\mathbf{E}^*$ for Π^* with arbitrary IV such that for all adversaries $\mathbf{D}$ there exist $\mathbf{D}_1$ and $\mathbf{D}_2$ such that*

$$\mathsf{PA1}_{\Pi^*}^{\mathbf{E}^*}(\mathbf{D}) \leq \mathsf{PA1}_{\Pi}^{\mathbf{E}}(\mathbf{D}_1) + \mathsf{PRF}_G(\mathbf{D}_2),$$

where $\mathbf{D}_1$ and $\mathbf{D}_2$ are as efficient as $\mathbf{D}$, and $\mathbf{E}^*$ is as efficient as $\mathbf{E}$.

Proof. Define $\mathbf{E}^*$ as follows. First $\mathbf{E}^*$ transforms its query history by mapping $\mathcal{E}_{K,K'}^*(IV, A, M) = (IV, A, C, T)$ to $\mathcal{E}_K(A, M) = (IV^*, A, C, T^*)$, where $T = IV^*T^*$ and $|IV^*| = n$. Then on input (IV, A, C, T), $\mathbf{E}^*$ returns $\mathbf{E}(IV^*, A, C, T^*)$, where again $T = IV^*T^*$ and $|IV^*| = n$.

The remainder of the proof is similar to the proof of Prop. 5. □

Note that the PRF-to-IV method is the basic structure behind SIV, BTM, and HBS. We show that the PRF-to-IV method is INT-RUP in Sect.7.2.

Online Encryption. Since the PRF needs to be computed over the entire message before the message is encrypted again, the PRF-to-IV method does not allow for online encryption. Recall that an encryption scheme has online encryption if for all (K, IV, A), the resulting function is online. Examples of such schemes include COPA and McOE-G.

If we want encryption and decryption to both be online in the arbitrary IV setting, then a large amount of ciphertext expansion is necessary, otherwise a distinguisher similar to the one used in the proof of Prop. 1 can be created.

An encryption scheme $\Pi = (\mathcal{E}, \mathcal{D})$ is online if for some n there exist functions f_i and f_i' such that

$$\mathcal{E}_K(M) = f_n(M_1)\, f_{2n}(M_1M_2) \cdots f_{jn}(M_1M_2 \cdots M_j)\, f'_{|M|}(M),$$

where $j = \lfloor (|M| - 1)/n \rfloor$ and M_i is the ith n-bit block of M. If the encryption scheme has online decryption as well, then the decryption algorithm can start decrypting each "block" of ciphertext, or

$$\mathcal{D}_K(f_n(M_1)\, f_{2n}(M_1M_2) \cdots f_{in}(M_1M_2 \cdots M_i)) = M_1M_2 \cdots M_i,$$

for all $i \leq j$.

Proposition 7. *Let $\Pi = (\mathcal{E}, \mathcal{D})$ be an encryption scheme where $\mathcal{E}$ is n-online for all K, IV, and A, and $\mathcal{D}$ is online as well, then there exists a PA1-adversary $\mathbf{D}$ such that for all extractors $\mathbf{E}$ there exists an IND-CPA adversary $\mathbf{D}_1$ such that*

$$1 - \mathsf{CPA}_{\Pi}(\mathbf{D}_1) \leq \mathsf{PA1}_{\Pi}^{\mathbf{E}}(\mathbf{D}) \ ,$$

where $\mathbf{D}$ makes one $\mathcal{O}_1$ query, one $\mathcal{O}_2$ query, and $\mathbf{D}_1$ is as efficient as $\mathbf{D}$ plus one query to $\mathbf{E}$.

Proof. Following Def. 2, we write

$$\mathcal{E}_K(M) = f_n(M_1)\, f_{2n}(M_1M_2) \cdots f_{jn}(M_1M_2 \cdots M_j)\, f'_{|M|}(M).$$

Since decryption is online as well, each f_{in} must be an injective function of M_i. Say that the ith block does not contain any length-expansion, i.e. for all $M \in \{0,1\}^m$, $|f_{in}(M_1 \cdots M_i)| = |M_i|$. Then f_{in} is bijective.

Let $C := f_n(M_1)\, f_{2n}(M_1M_2) \cdots f_{(i-1)n}(M_1 \cdots M_{i-1})$, then for all K, C_i in the range of f_{in}, and C' we have

$$\mathcal{D}_K(CC_iC') = M_1 \cdots M_{i-1}M_iM',$$

and so

$$\begin{aligned}\mathcal{E}_K(M_1 \cdots M_{i-1}M_iM') &= f_n(M_1) \cdots f_{(i-1)n}(M_1 \cdots M_{i-1}) f_{in}(M_i)C'' \\ &= CC_iC''\end{aligned}$$

or

$$\mathcal{E}_K(\mathcal{D}_K(CC_iC')) = CC_iC'' \ . \tag{2}$$

Equation (2) always holds in the real world, but is difficult to simulate by the plaintext extractor. Therefore we define the adversary $\mathbf{D}$ as follows. The adversary picks C_i uniformly at random from the range of f_{in}. It queries $\mathcal{O}_1(M)$ for some message M of length $(i-1)n$ and receives C_0 from which it chops off bits from the end to create a string C of length $(i-1)n$. Then $\mathbf{D}$ queries $\mathcal{O}_2(CC_i) = M'$ and finally $\mathcal{O}_1(M') = C'$. If C' has CC_i as a prefix then $\mathbf{D}$ outputs 1, otherwise it outputs 0.

As in the proof of Prop. 1, any extractor which successfully finds an M_i such that CC_i is a prefix for $\mathcal{E}_K(MM_i)$ has broken the IND-CPA game for Π. □

Example 2. In certain scenarios, padding the plaintext is sufficient for PA1. Doing so makes schemes such as McOE-G secure in the sense of PA1, while keeping encryption and decryption online. The cost is a substantial expansion of the ciphertext. For the case of McOE-G, the length of the ciphertext becomes roughly twice the size of its plaintext.

It is important to note that McOE-G is based on an n-bit block cipher, and each n-bit message block is encrypted (after it is XORed with some state values) via the block cipher call. Since the underlying block cipher is assumed to be a strong pseudo-random function (SPRP), we can pad a message $M = M_1M_2 \cdots M_\ell$ (each M_i is an $n/2$-bit string) as $0^{n/2}M_1 \,\|\, 0^{n/2}M_2 \,\|\, \cdots \,\|\, 0^{n/2}M_\ell$ and then encrypt this padded message using McOE-G. So each block cipher call processes $0^{n/2}M_i$ for some i. This "encode-then-encipher" scheme [12] is PA1 as shown in App. B.

Example 3. If we do not require the decryption to be online, then we can achieve PA1 without significant ciphertext expansion. An example of a scheme that falls into this category is the recently-introduced APE mode [2], whose decryption is backward (and hence not online). See App. C for a statement and proof.

6.5 PA2 Schemes

Most AE schemes are proven to be IND-CPA and INT-CTXT, which allows one to achieve IND-CCA [9] assuming verification works correctly. In order to be as efficient as possible, the underlying encryption schemes in the AE schemes are designed to only achieve IND-CPA and not IND-CCA, since achieving IND-CCA for encryption usually requires significantly more operations. For example, GCM, SIV, BTM, and HBS all use CTR mode for encryption, yet CTR mode is not IND-CCA. Since IND-CPA+PA2 is equivalent to IND-CCA′, none of these schemes achieve PA2.

A scheme such as APE also cannot achieve IND-CCA′ because its decryption is online "in reverse". If $(\mathcal{E}_K, \mathcal{D}_K)$ denotes APE, then an adversary can query $\mathcal{E}_K(M_1M_2) = C_1C_2$ and then $\mathcal{D}_K(C_1'C_2)$, which equals $M_1'M_2$. But if an adversary interacts with $(\$, \mathcal{D}_K)$ (see Def. 3), then $\mathcal{D}_K(C_1'C_2)$ will most likely not output $M_1'M_2$.

Existing designs which do achieve PA2 include those which are designed to be IND-CCA′, such as the solutions presented by Bellare and Rogaway [12], Desai [21], and Shrimpton and Terashima [41]. These solutions cannot be online, and they are usually at least "two-pass", meaning the input is processed at least twice.

7 Integrity in the INT-RUP Setting

7.1 INT-RUP Attack

Several AE schemes become insecure if unverified plaintext is released. In Proposition 8, we explain that OCB [37] and COPA [3] are not secure in the RUP setting.

The strategy of our attack is similar to that of Bellare and Micciancio on the XHASH hash function [8]. However, our attack is an improved version that solves a system of linear equations in $GF(2)$ with only half the number of equations and variables. Our attack is also related to the attack by Saarinen [40] on the FSB [4] hash function, and the attack on the hash function Maraca [29] by Indesteege and Preneel [26].

The attack works by first querying the encryption oracle under nonce N to get a valid ciphertext and tag pair. Then, two decryption queries are made under the same nonce N. Using the resulting plaintexts a system of linear equations is set up, which when solved will give the a forgery with high probability.

Proposition 8. *For OCB and COPA, for all $\ell \geq n$ there exists an adversary $\mathbf{A}$ such that*

$$\mathsf{INT\text{-}RUP}_{\Pi}(\mathbf{A}) \geq 1 - 2^{n-\ell} \ ,$$

where $\mathbf{A}$ makes one encryption query and two decryption queries, each consisting of ℓ blocks of n bits. Then, the adversary solves a system of linear equations in $GF(2)$ with n equations and ℓ unknowns.

Proof. We start by describing OCB for messages which have a length which is a multiple of the block size. For our purposes it suffices to describe OCB in terms of families of ideal permutations, as the attack works when the block cipher is replaced by a random permutation (a random permutation is a permutation chosen uniformly from a finite set of permutations).

Let $\Pi = (\mathcal{E}, \mathcal{D}, \mathcal{V})$ denote OCB operating only on full message blocks. Let $\{\alpha_i^N, \beta_i^N, \gamma_i^N\}$ be independent random permutations with domain $\{0,1\}^n$ and range $\{0,1\}^n$, then

$$\mathcal{E}_K(N, M_1M_2\cdots M_\ell) = (N, C_1C_2\cdots C_\ell, T) \ ,$$

where

$$\begin{aligned} C_i &= \alpha_i^N(M_i) \qquad \text{for } 1 \leq i < \ell \ , \\ C_\ell &= \beta_\ell^N(\mathrm{len}(n)) \oplus M_\ell \ , \\ T &= \gamma_\ell^N\Big(M_1 \oplus \cdots \oplus M_\ell\Big) \ , \end{aligned}$$

and $\mathrm{len}(n)$ is the number n represented as an n-bit string.

Given a valid plaintext-ciphertext pair, our attack makes two queries to the decryption oracle, and then solves a system of linear equations in $GF(2)$ in order to obtain a forgery.

Let $\ell \geq n$. Firstly, the adversary queries $\mathcal{E}_K(N, M) = (N, C, T)$ where $M = M_1 M_2 \cdots M_\ell$ consists of ℓ blocks of n bits, and N is some fixed value. Let $C = C_1 C_2 \cdots C_\ell$ and let $Z = M_1 \oplus \cdots \oplus M_\ell$.

If the adversary can create another plaintext M' with the same checksum Z by changing the message blocks $M_1, M_2, ..., M_\ell$, it has constructed a forgery because the checksum Z and therefore the tag T will be the same. The adversary is not allowed to query two encryptions under the same nonce N. However, we now show that it is possible to construct a forgery by querying the decryption oracle twice with the same nonce N and observing the unverified plaintext.

The adversary now chooses $C^0 = C_1^0 C_2^0 \cdots C_\ell^0 T^0$ and $C^1 = C_1^1 C_2^1 \cdots C_\ell^1 T^1$ uniformly at random such that for each i, C_i^0, C_i^1, C_i are all distinct. The corresponding unverified plaintexts are $\mathcal{D}_K(N, C^0, T^0) = M_1^0 M_2^0 \cdots M_\ell^0$ and $\mathcal{D}_K(N, C^1, T^1) = M_1^1 M_2^1 \cdots M_\ell^1$. To construct a plaintext $M' = M_1^{x_1} M_2^{x_2} \cdots M_\ell^{x_\ell}$ with the same checksum as M, the adversary has to find $x_1, x_2, \cdots, x_\ell \in \mathrm{GF}(2)$ such that

$$Z = \oplus_{i=1}^{\ell} \big(M_i^0 x_i \oplus M_i^1 (x_i \oplus 1) \big) \ ,$$

where $x_i = 1$ corresponds to selecting M_i^0, and $x_i = 0$ to selecting M_i^1 as the ith message block of M'. This expression can be converted into n equations, one for every bit j:

$$Z[j] = \oplus_{i=1}^{\ell} \big(M_i^0[j] x_i \oplus M_i^1[j](x_i \oplus 1) \big) \qquad \text{for } j = 0, 1, \ldots, n-1 \ ,$$

where $X[j]$ selects jth bit of X, with $j = 0$ corresponding to the least significant bit.

This is a system of linear equations in $GF(2)$ with n equations and ℓ unknowns, for which a solution can be found using Gaussian elimination. The probability that this system of equations has a solution, is at least $1 - 2^{n-\ell}$ [8, App. A]. Because $\mathcal{E}_K(N, M') = (N, C', T)$ with $C' = C_1^{x_1} C_2^{x_2} \cdots C_\ell^{x_\ell}$ and $C' \neq C$, the adversary can output (N, C', T) as a forgery.

Observe that for COPA [3], the tag is also generated as the XOR of the message blocks. Therefore, the same attack strategy that we described for OCB in the INT-RUP setting also applies to COPA. □

7.2 Nonce Decoy and PRF-to-IV

In Sect. 6 we introduced a way of turning a random IV PA1 scheme into a nonce IV PA1 scheme, the nonce decoy, and a way of turning a random IV PA1 scheme into an arbitrary IV PA1 scheme, the PRF-to-IV method. Here we consider what happens to INT-RUP when the two methods are applied.

The nonce decoy adds some integrity to the underlying random IV PA1 scheme. Using the notation from Sect. 6.3, Π needs to be a slightly lighter form of INT-RUP in order for Π^* to be INT-RUP. Concretely, Π only needs to be INT-RUP against adversaries which use IVs which are the result of an encryption query. Furthermore, this requirement on Π is sufficient to prove that Π^* is INT-RUP.

Proposition 9. *Let* $\mathbf{D}$ *be an adversary which only makes queries to* $\mathcal{V}_K$ *with an IV which is the result of an* $\mathcal{E}_K$ *query. Then there exist adversaries* $\mathbf{D}_1$ *and* $\mathbf{D}_2$ *such that*

$$\mathsf{INT\text{-}RUP}_{\Pi}(\mathbf{D}) \leq \mathsf{INT\text{-}RUP}_{\Pi^*}(\mathbf{D}_1) + \mathsf{PRF}_G(\mathbf{D}_2),$$

where $\mathbf{D}_1$ *and* $\mathbf{D}_2$ *are as efficient as* $\mathbf{D}$.

Proof. The adversary $\mathbf{D}_1$ runs $\mathbf{D}$ and answers $\mathbf{D}$'s oracle queries as follows. The $\mathcal{E}_K$ and $\mathcal{D}_K$ oracle queries are dealt with like in the proof of Prop. 3. On a $\mathcal{V}_K(IV, A, C, T)$ query, $\mathbf{D}_1$ responds with $\mathcal{V}^*_{K,K'}(i, A, IV \| C, T)$, where i equals the counter value where $\mathcal{E}^*_{K,K'}(i, A', M') = (IV, A', C', T')$. □

Naturally if Π is INT-RUP, then Π^* is INT-RUP as well. In fact, if Π is INT-RUP against adversaries which use IVs which are the result of an encryption query, then Π^* is INT-RUP

Proposition 10. *Let* $\mathbf{D}$ *be an INT-RUP adversary for* Π^* *with random IV. Then there exists an INT-RUP adversary* $\mathbf{D}_1$ *for* Π *which only uses IVs from encryption queries, and an adversary* $\mathbf{D}_2$ *such that*

$$\mathsf{INT\text{-}RUP}_{\Pi^*}(\mathbf{D}) \leq \mathsf{INT\text{-}RUP}_{\Pi}(\mathbf{D}_1) + \mathsf{PRF}_G(\mathbf{D}_2) \ ,$$

where $\mathbf{D}_1$ *and* $\mathbf{D}_2$ *are as efficient as* $\mathbf{D}$.

Proof. We transform $\mathbf{D}$'s queries into the appropriate way as in the previous proofs, and notice that if $\mathbf{D}$ uses an IV which is not the output of an encryption query and successfully forges, then $\mathbf{D}$ has an attack on G. □

The PRF-to-IV method is a much stronger transform than the nonce decoy. Following the notation from Sect. 6.4, we do not need to assume anything about the underlying random IV scheme Π in order to prove that Π^* is INT-RUP.

Proposition 11. *Let* $\mathbf{D}$ *be an INT-RUP adversary for* Π^*. *Then there exists an adversary* $\mathbf{D}_1$ *such that*

$$\mathsf{INT\text{-}RUP}_{\Pi^*}(\mathbf{D}) \leq \mathsf{PRF}_G(\mathbf{D}_1),$$

where $\mathbf{D}_1$ *is as efficient as* $\mathbf{D}$.

Proof. The adversary $\mathbf{D}_1$ runs $\mathbf{D}$, generates a random key K, and simulates $\mathcal{E}^*_{K,K'}$, $\mathcal{D}^*_{K,K'}$, and $\mathcal{V}^*_{K,K'}$ using K and its own oracle. □

8 Another Look at the Privacy and Integrity Notions

Typically, the security of an AE scheme is examined under IND-CPA and INT-CTXT regarding privacy and integrity, respectively. IND-CPA + INT-CTXT can alternatively be integrated into of a single conventional AE security notion defined as $\Delta_{\mathbf{D}}(\mathcal{E}_K, \mathcal{V}_K\,;\, \$, \perp)$ under our new syntax.

In this work, however, we examine the RUP situation where an AE adversary is endowed with extended capabilities, namely observing the results of ciphertext decryptions and this we model by giving the adversary additional access to a decryption only oracle. To capture AE security under RUP we proposed the notions of IND-CPA+PA1 or IND-CPA+PA2 for the privacy part, and INT-RUP for the integrity part. Overall, the security target for an AE scheme in our extended RUP model becomes the fusion of these three notions IND-CPA + PA1/PA2 + INT-RUP.

In what follows we give a notion AE-RUP1 which we show is exactly the gap between conventional AE security and AE security in the RUP setting.

We introduce AE-RUP1 as a measure of the security loss resulting from the adversarial extra access to $\mathcal{D}_K$ in the RUP setting:

$$\mathsf{AE\text{-}RUP1}_{\Pi}(\mathbf{D}) := \underset{\mathbf{D}}{\Delta}(\mathcal{E}_K, \mathcal{D}_K, \mathcal{V}_K\,;\, \mathcal{E}_K, \mathbf{E}, \mathcal{V}_K) \ ,$$

where $\mathbf{E}$ is the extractor used in the PA1 definition. As we show, similar results can be obtained for AE-RUP2, where the extractor of the PA2 definition is used instead.

To understand why AE-RUP1 is the exact measure of the security gap that occurs between the conventional and RUP settings, we need to examine the impact of AE-RUP1 in its combination with the integrity INT-CTXT notion.

- **INT-CTXT + AE-RUP1 ⇒ INT-RUP + PA1.**
  We recall that INT-CTXT is defined as $\Delta_{\mathbf{D}}(\mathcal{E}_K, \varnothing, \mathcal{V}_K\,;\, \mathcal{E}_K, \varnothing, \perp)$. Next, INT-RUP can be defined as $\Delta_{\mathbf{D}}(\mathcal{E}_K, \mathcal{D}_K, \mathcal{V}_K\,;\, \mathcal{E}_K, \mathcal{D}_K, \perp)$. Because INT-CTXT + AE-RUP1 ⇒ INT-RUP and AE-RUP1 ⇒ PA1 are direct results from the definition of AE-RUP1, the statement follows.

– **INT-CTXT + AE-RUP1 ⇐ INT-RUP + PA1.**
It is straightforward that INT-RUP ⇒ INT-CTXT. Also, INT-RUP+PA1 ⇒ AE-RUP1 is a direct result from the respective definitions. The statement then follows.

As a result we get IND-CPA + INT-CTXT + AE-RUP1 ⇔ IND-CPA + PA1 + INT-RUP. Using the same proof strategy, but changing the definition of the extractor, we have that IND-CPA + INT-CTXT + AE-RUP2 ⇔ IND-CPA + PA2 + INT-RUP.

9 Conclusions

In many practical applications, it is desirable that an AE scheme can securely output plaintext before verification. We formalized this security under the release of unverified plaintext (RUP) by separating decryption and verification.

Two notions of plaintext awareness (PA1 and PA2) were introduced to symmetric cryptosystems. In the RUP setting, privacy is achieved as a combination of IND-CPA and PA1 or PA2. For integrity, we introduced the INT-RUP notion as an extension of INT-CTXT, where a forger may abuse unverified plaintext. We connected our notions of privacy and integrity in the RUP setting to existing security notions, and saw that the relations and separations depended on the IV type.

While the CTR and CBC modes with a random IV achieve IND-CPA+PA1, this turns out to be non-trivial for nonce-based or deterministic encryption schemes. Our results showed that many AE schemes such as GCM, CCM, COPA, and McOE-G are not secure in the RUP setting. We provided remedies for both nonce-based and deterministic AE schemes. For the former case, we introduced the *nonce decoy* technique, which allowed to transform a nonce to a random-looking IV. The PRF-to-IV method converts random IV PA1 schemes into arbitrary IV PA1 schemes. We showed that deterministic AE schemes cannot be PA1, unless the decryption is offline (as in APE) or there is significant ciphertext expansion.

Future Work. Given that our PRF-to-IV method is rather inefficient, we leave it as an open problem to efficiently modify any encryption-only scheme into an AE scheme that is INT-RUP. A related problem is to fix OCB and COPA to be INT-RUP in an efficient way. The PA1 solutions we provide all start with the assumption that the nonce IV or arbitrary IV scheme is PA1 when a random IV is used instead. An interesting problem is to find alternative solutions to constructing nonce IV and arbitrary IV PA1 schemes. A problem of theoretical interest is to find a non-pathological random IV encryption scheme that is not PA1. In some applications, formalizing security in the RUP setting as IND-CPA+PA1 and INT-RUP may be sufficient. It is interesting to investigate how well this formalization reflects the problems encountered in real-world implementations, to see where PA2 may also be necessary.

Acknowledgments. This work was supported in part by the Research Council KU Leuven: GOA TENSE (GOA/11/007) and OT/13/071. Elena Andreeva and Nicky Mouha are Postdoctoral Fellows of the Research Foundation – Flanders (FWO). Atul Luykx is supported by a Ph.D. Fellowship from the Institute for the Promotion of Innovation through Science and Technology in Flanders (IWT-Vlaanderen).

References

1. AlFardan, N.J., Paterson, K.G.: Lucky Thirteen: Breaking the TLS and DTLS Record Protocols. In: IEEE Symposium on Security and Privacy. pp. 526–540. IEEE Computer Society (2013)
2. Andreeva, E., Bilgin, B., Bogdanov, A., Luykx, A., Mennink, B., Mouha, N., Yasuda, K.: APE: Authenticated Permutation-Based Encryption for Lightweight Cryptography. In: Cid, S., Rechberger, C. (eds.) FSE 2014. Lecture Notes in Computer Science, Springer (2014), Available at: `http://eprint.iacr.org/2013/791`

3. Andreeva, E., Bogdanov, A., Luykx, A., Mennink, B., Tischhauser, E., Yasuda, K.: Parallelizable and Authenticated Online Ciphers. In: Sako, K., Sarkar, P. (eds.) ASIACRYPT. Lecture Notes in Computer Science, Springer (2013)
4. Augot, D., Finiasz, M., Sendrier, N.: A Family of Fast Syndrome Based Cryptographic Hash Functions. In: Dawson, E., Vaudenay, S. (eds.) Mycrypt. Lecture Notes in Computer Science, vol. 3715, pp. 64–83. Springer (2005)
5. Bellare, M., Desai, A., Jokipii, E., Rogaway, P.: A Concrete Security Treatment of Symmetric Encryption. In: FOCS. pp. 394–403. IEEE Computer Society (1997)
6. Bellare, M., Desai, A., Pointcheval, D., Rogaway, P.: Relations Among Notions of Security for Public-Key Encryption Schemes. In: Krawczyk, H. (ed.) CRYPTO. Lecture Notes in Computer Science, vol. 1462, pp. 26–45. Springer (1998)
7. Bellare, M., Keelveedhi, S.: Authenticated and Misuse-Resistant Encryption of Key-Dependent Data. In: Rogaway, P. (ed.) CRYPTO 2011. Lecture Notes in Computer Science, vol. 6841, pp. 610–629. Springer (2011)
8. Bellare, M., Micciancio, D.: A New Paradigm for Collision-Free Hashing: Incrementality at Reduced Cost. In: Fumy, W. (ed.) EUROCRYPT. Lecture Notes in Computer Science, vol. 1233, pp. 163–192. Springer (1997)
9. Bellare, M., Namprempre, C.: Authenticated Encryption: Relations among Notions and Analysis of the Generic Composition Paradigm. In: Okamoto, T. (ed.) ASIACRYPT 2000. Lecture Notes in Computer Science, vol. 1976, pp. 531–545. Springer (2000)
10. Bellare, M., Palacio, A.: Towards Plaintext-Aware Public-Key Encryption Without Random Oracles. In: Lee, P.J. (ed.) ASIACRYPT. Lecture Notes in Computer Science, vol. 3329, pp. 48–62. Springer (2004)
11. Bellare, M., Rogaway, P.: Optimal Asymmetric Encryption. In: Santis, A.D. (ed.) EUROCRYPT. Lecture Notes in Computer Science, vol. 950, pp. 92–111. Springer (1994)
12. Bellare, M., Rogaway, P.: Encode-Then-Encipher Encryption: How to Exploit Nonces or Redundancy in Plaintexts for Efficient Cryptography. In: Okamoto, T. (ed.) ASIACRYPT. Lecture Notes in Computer Science, vol. 1976, pp. 317–330. Springer (2000)
13. Bellare, M., Rogaway, P.: Introduction to modern cryptography. In: UCSD CSE 207 Course Notes (September 2005)
14. Bellare, M., Rogaway, P.: The security of triple encryption and a framework for code-based game-playing proofs. In: Vaudenay, S. (ed.) EUROCRYPT. Lecture Notes in Computer Science, vol. 4004, pp. 409–426. Springer (2006)
15. Bernstein, D.J.: Features of various secret-key primitives (January 2014), `http://competitions.cr.yp.to/features.html`
16. Bertoni, G., Daemen, J., Peeters, M., Assche, G.V.: Duplexing the Sponge: Single-Pass Authenticated Encryption and Other Applications. In: Miri, A., Vaudenay, S. (eds.) Selected Areas in Cryptography 2011. Lecture Notes in Computer Science, vol. 7118, pp. 320–337. Springer (2012)
17. Bogdanov, A., Mendel, F., Regazzoni, F., Rijmen, V., Tischhauser, E.: ALE: AES-Based Lightweight Authenticated Encryption. In: Moriai [32]
18. Boldyreva, A., Degabriele, J.P., Paterson, K.G., Stam, M.: Security of Symmetric Encryption in the Presence of Ciphertext Fragmentation. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. Lecture Notes in Computer Science, vol. 7237, pp. 682–699. Springer (2012)
19. Boldyreva, A., Degabriele, J.P., Paterson, K.G., Stam, M.: On Symmetric Encryption with Distinguishable Decryption Failures. In: Moriai [32]
20. Canvel, B., Hiltgen, A.P., Vaudenay, S., Vuagnoux, M.: Password Interception in a SSL/TLS Channel. In: Boneh, D. (ed.) CRYPTO. Lecture Notes in Computer Science, vol. 2729, pp. 583–599. Springer (2003)
21. Desai, A.: New Paradigms for Constructing Symmetric Encryption Schemes Secure against Chosen-Ciphertext Attack. In: Bellare, M. (ed.) CRYPTO. Lecture Notes in Computer Science, vol. 1880, pp. 394–412. Springer (2000)
22. Dolev, D., Dwork, C., Naor, M.: Nonmalleable Cryptography. SIAM J. Comput. 30(2), 391–437 (2000)
23. Fleischmann, E., Forler, C., Lucks, S.: McOE: A Family of Almost Foolproof On-Line Authenticated Encryption Schemes. In: Canteaut, A. (ed.) FSE. Lecture Notes in Computer Science, vol. 7549, pp. 196–215. Springer (2012)
24. Fouque, P.A., Joux, A., Martinet, G., Valette, F.: Authenticated On-Line Encryption. In: Matsui, M., Zuccherato, R.J. (eds.) Selected Areas in Cryptography. Lecture Notes in Computer Science, vol. 3006, pp. 145–159. Springer (2003)
25. Goldwasser, S., Micali, S.: Probabilistic Encryption and How to Play Mental Poker Keeping Secret All Partial Information. In: Lewis, H.R., Simons, B.B., Burkhard, W.A., Landweber, L.H. (eds.) STOC 1982. pp. 365–377. ACM (1982)
26. Indesteege, S., Preneel, B.: Practical Preimages for Maraca. In: Proceedings of the 30th Symposium on Information Theory in the Benelux. pp. 119–126 (2009)
27. Iwata, T., Yasuda, K.: BTM: A Single-Key, Inverse-Cipher-Free Mode for Deterministic Authenticated Encryption. In: Jr., M.J.J., Rijmen, V., Safavi-Naini, R. (eds.) Selected Areas in Cryptography. Lecture Notes in Computer Science, vol. 5867, pp. 313–330. Springer (2009)

28. Iwata, T., Yasuda, K.: HBS: A Single-Key Mode of Operation for Deterministic Authenticated Encryption. In: Dunkelman, O. (ed.) FSE. Lecture Notes in Computer Science, vol. 5665, pp. 394–415. Springer (2009)
29. Jenkins, Jr., R.J.: Algorithm Specification. Submission to NIST (2008), http://burtleburtle.net/bob/crypto/maraca/nist/Supporting_Documentation/specification.pdf
30. Krovetz, T., Rogaway, P.: The OCB Authenticated-Encryption Algorithm. http://datatracker.ietf.org/doc/draft-irtf-cfrg-ocb (June 2013)
31. McGrew, D.A., Viega, J.: The Security and Performance of the Galois/Counter Mode (GCM) of Operation. In: Canteaut, A., Viswanathan, K. (eds.) INDOCRYPT. Lecture Notes in Computer Science, vol. 3348, pp. 343–355. Springer (2004)
32. Moriai, S. (ed.): Fast Software Encryption - 20th International Workshop, FSE 2013, Singapore, March 10-13, 2013. Revised Selected Papers. Lecture Notes in Computer Science, Springer (2013)
33. National Institute of Standards and Technology: DES Modes of Operation. FIPS 81 (December 1980)
34. Paterson, K.G., AlFardan, N.J.: Plaintext-Recovery Attacks Against Datagram TLS. In: NDSS. The Internet Society (2012)
35. Rogaway, P.: Authenticated-encryption with associated-data. In: Atluri, V. (ed.) ACM Conference on Computer and Communications Security 2002. pp. 98–107. ACM (2002)
36. Rogaway, P.: Nonce-Based Symmetric Encryption. In: Roy, B.K., Meier, W. (eds.) FSE 2004. Lecture Notes in Computer Science, vol. 3017, pp. 348–359. Springer (2004)
37. Rogaway, P., Bellare, M., Black, J.: OCB: A Block-Cipher Mode of Operation for Efficient Authenticated Encryption. ACM Trans. Inf. Syst. Secur. 6(3), 365–403 (2003)
38. Rogaway, P., Shrimpton, T.: A Provable-Security Treatment of the Key-Wrap Problem. In: Vaudenay, S. (ed.) EUROCRYPT 2006. Lecture Notes in Computer Science, vol. 4004, pp. 373–390. Springer (2006)
39. Rogaway, P., Wagner, D.: A Critique of CCM. Cryptology ePrint Archive, Report 2003/070 (2003)
40. Saarinen, M.J.O.: Linearization Attacks Against Syndrome Based Hashes. In: Srinathan, K., Rangan, C.P., Yung, M. (eds.) INDOCRYPT. Lecture Notes in Computer Science, vol. 4859, pp. 1–9. Springer (2007)
41. Shrimpton, T., Terashima, R.S.: A modular framework for building variable-input-length tweakable ciphers. In: Sako, K., Sarkar, P. (eds.) ASIACRYPT (I). Lecture Notes in Computer Science, vol. 8269, pp. 405–423. Springer (2013)
42. Tsang, P.P., Smith, S.W.: Secure cryptographic precomputation with insecure memory. In: Chen, L., Mu, Y., Susilo, W. (eds.) ISPEC 2008. Lecture Notes in Computer Science, vol. 4991, pp. 146–160. Springer (2008)
43. Tsang, P.P., Solomakhin, R.V., Smith, S.W.: Authenticated streamwise on-line encryption. Dartmouth Computer Science Technical Report TR2009-640 (2009)
44. Vaudenay, S.: Security Flaws Induced by CBC Padding - Applications to SSL, IPSEC, WTLS ... In: Knudsen, L.R. (ed.) EUROCRYPT. Lecture Notes in Computer Science, vol. 2332, pp. 534–546. Springer (2002)
45. Whiting, D., Housley, R., Ferguson, N.: Counter with CBC-MAC (CCM). Request For Comments 3610 (2003)
46. Wu, H., Preneel, B.: AEGIS – An Efficient Authenticated Encryption Algorithm. In: Lange, T., Lauter, K., Lisonek, P. (eds.) Selected Areas in Cryptography. Lecture Notes in Computer Science, Springer (2013)

A Random IV CTR and CBC Modes

CBC Mode Description. Let $E : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ be an SPRP. For $M_i \in \{0,1\}^n$, $1 \le i \le \ell$, define RIV-CBC encryption as $\mathcal{E}_K(M_1 \cdots M_\ell) = C_1 \cdots C_\ell$, where C_0 is selected uniformly at random from $\{0,1\}^n$ for each encryption, and

$$C_i = E_K(C_{i-1} \oplus M_i) \text{ for } 1 \le i \le \ell \ .$$

Note that we have suppressed the explicit IV input to $\mathcal{E}_K$ and use C_0 instead. Decryption is similarly defined as $\mathcal{D}_K(C_0, C_1 \cdots C_\ell) = M_1 \cdots M_\ell$, where

$$M_i = E_K^{-1}(C_i) \oplus C_{i-1} \text{ for } 1 \le i \le \ell \ .$$

We can define an extractor $\mathbf{E}$ for RIV-CBC as follows. Initially, $\mathbf{E}$ generates a random key K' which it will use via $F_{K'}$. Let $(C_0, C_1 \cdots C_\ell)$ denote an input to $\mathbf{E}$. For all $1 \le i \le \ell$, the extractor operates as follows: it searches its history for two adjacent ciphertext blocks $(C'_{i-1}, C'_i) = (C_{i-1}, C_i)$. If such couple exists, it defines M_i to be the corresponding M'_i. Otherwise, it generates $M_i = E_{K'}^{-1}(C_i) \oplus C_{i-1}$.

Proposition 12. *Let* $\mathbf{D}$ *be a PA1 adversary for RIV-CTR or RIV-CBC making queries whose lengths in number of blocks sum up to* σ*, then*

$$\mathsf{PA1}^{\mathbf{E}}_{\mathit{RIV\text{-}CTR}}(\mathbf{D}) \leq \underset{\mathbf{D}_1}{\Delta}(F_K, F_K\,;\, F_K, F_{K'}) + \frac{\sigma^2}{2^n}\ ,$$

or

$$\mathsf{PA1}^{\mathbf{E}}_{\mathit{RIV\text{-}CBC}}(\mathbf{D}) \leq \underset{\mathbf{D}_1}{\Delta}(E_K, E_K^{-1}\,;\, E_K, E_{K'}^{-1}) + \frac{\sigma^2}{2^n - \sigma}\ ,$$

where $\mathbf{D}_1$ *is an adversary which may not make the same query to both of its oracles, and makes a total of* σ *queries with the same running time as* $\mathbf{D}$*.*

Proof. We start with the proof for RIV-CTR. First we note that $\mathbf{E}$ behaves exactly the same as $\mathcal{D}_K$ unless it receives a query $C_0 \cdots C_m$ where

1. C_0 has been returned as an IV by a previous query to $\mathcal{E}_K$ and m is larger than the number of blocks in all such queries, or
2. C_0 has never been returned as an IV by a previous query to $\mathcal{E}_K$.

In both cases $\mathbf{E}$ must make queries to $F_{K'}$ in order to generate randomness.

We construct adversary $\mathbf{D}_1$ which copies $\mathcal{E}_K$ exactly with its first oracle and mimics $\mathbf{E}$ with a combination of its first and its second oracle. Observe that $\mathbf{E}$ behaves exactly like $\mathcal{D}_K$ if $K = K'$. As a result, we almost get a perfect simulation of PA1. The only problem is when $\mathbf{D}$ induces $\mathbf{D}_1$ to make the same query on both its oracles; then we violate the definition of $\mathbf{D}_1$ (and $\mathbf{D}_1$ would be able to trivially win its game). The only time in which $F_{K'}$ is called is in the cases listed above. Hence the probability that F_K and $F_{K'}$ are queried on the same input is upper bounded by the probability that the sequences of strings generated starting from the IVs collide with each other: each $\mathcal{E}_K$ query leads to a sequence of F_K queries starting from the IV, and each $\mathcal{D}_K$ could lead to a sequence of $F_{K'}$ queries starting from C_0. Since the IVs are generated independently and uniformly at random, this probability is upper bounded by $\sigma^2/2^n$, where σ is the sum of the number of blocks of all the F-queries made by $\mathbf{D}'$.

Now, for RIV-CBC we consider an SPRP E_K and the response to every invocation is drawn from a set of size at least $2^n - \sigma$ instead of 2^n. The remainder of the proof is in essence the same, and we skip the details. □

B Adjusted McOE-G is PA1

Proposition 13. *Let* $\mathbf{D}$ *be a PA1 adversary for the adjusted McOE-G (called aMcOE-G) making queries whose lengths in number of blocks sum up to* σ*, then*

$$\mathsf{PA1}^{\mathbf{E}}_{\mathit{aMcOE\text{-}G}}(\mathbf{D}) \leq \frac{\sigma}{2^{n/2}} + \frac{\sigma^2}{2^{n+1}}\ .$$

Proof. We consider the McOE scheme using n-bit ideal tweakable ciphers E_t with tweaks t, applying the zero-padding $0^{n/2}M_1 \| 0^{n/2}M_2 \| \cdots \| 0^{n/2}M_w$ to a message M (so it is divided into $n/2$-bit message blocks) as shown in Fig. 3:

$\underline{\mathcal{E}(IV, M)}$:
$\tau \leftarrow E_0(IV)$
$t_1 \leftarrow IV \oplus \tau$
for $i = 1$ to w **do**
  $C_i \leftarrow E_{t_i}(0^{n/2}M_i)$
  $t_{i+1} \leftarrow 0^{n/2}M_i \oplus C_i$
end for

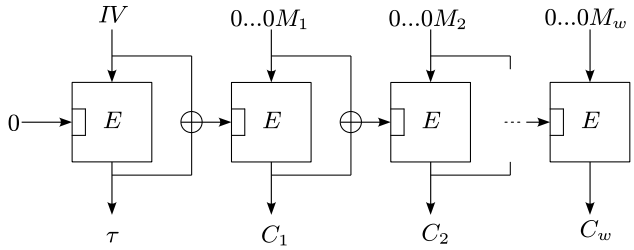

Fig. 3. The encryption part of the McOE Scheme with the zero padding calling an ideal tweakable cipher E.

To show that this scheme is PA1, we can construct a plaintext extractor $\mathbf{E}$ as follows. The extractor $\mathbf{E}$ receives a ciphertext $C_1C_2\cdots C_w$ and a tag from adversary $\mathbf{D}$. The extractor $\mathbf{E}$ then checks in the query history $\mathcal{Q}$ and its own history $\mathcal{S}$ for the longest common prefix. Say $\mathbf{E}$ has found the prefix $C_1C_2\cdots C_j$. Then $\mathbf{E}$ returns the corresponding plaintext blocks $M_1M_2\cdots M_j$ plus randomly generated suffix $M_{j+1}\cdots M_w$.

There are only two kinds of events, $\mathsf{E1}$ and $\mathsf{E2}$, when the extractor fails to mimic the decryption oracle. The event $\mathsf{E1}$ is when the adversary $\mathbf{D}$ makes a query containing a new block C_i (new relative to the tweak determined by the prefix) whose decrypted block happens to be of the form $0^{n/2}\|*$. The probability of this event happening is at most $\sigma/2^{n/2}$, where σ is the query complexity (in blocks) of the adversary. The other event $\mathsf{E2}$ is when a collision of tweak values occurs via the decryption oracle. Note that the extractor $\mathbf{E}$ does not take into account such an event. The probability of the event $\mathsf{E2}$ is at most $\binom{\sigma}{2}/2^n \le 0.5\sigma^2/2^n$. Overall, the advantage can be bounded as

$$\mathsf{PA1}^{\mathbf{E}}_{\mathcal{E}}(\mathbf{D}) \le \frac{\sigma}{2^{n/2}} + \frac{0.5\sigma^2}{2^n} \enspace .$$

□

C APE is PA1

Proposition 14. *Let* $\mathbf{D}$ *be a PA1 adversary for APE making queries whose lengths in number of blocks sum up to* σ*, then*

$$\mathsf{PA1}^{\mathbf{E}}_{APE}(\mathbf{D}) \le \frac{\sigma^2}{2^{r+c}} + \frac{2\sigma(\sigma+1)}{2^c} \enspace .$$

Proof. Let $\Pi = (\mathcal{E}, \mathcal{D}, \mathcal{V})$ denote the APE authenticated encryption scheme. $\mathcal{E}$ for integral associated data and message blocks is given in Fig. 4; we refer to [2] for a formal specification.

The extractor $\mathbf{E}$ for Π operates as follows. Let $(A, C_1\cdots C_\ell, T)$ denote an input of $\mathbf{E}$. For the tag T, let $(C'_1\cdots C'_{j-1}C_j\cdots C_{\ell'}, M'_1, \cdots M'_{\ell'})$ denote the query-response pair (to either $\mathcal{E}_K$ or $\mathbf{E}$) with the largest overlapping suffix in C. $\mathbf{E}$ sets $M_i = M'_{\ell'-\ell+i}$ for $i = \ell, \ldots, j+1$ and $M_j = M'_{\ell'-\ell+j} \oplus C'_{\ell'-\ell+j-1} \oplus C_{j-1}$ (where $C_0 = C'_0 = IV_r$ by definition), and generates $M_{j-1}, \ldots, M_1 \xleftarrow{R} \mathbf{R}$.

We focus on information-theoretic adversaries $\mathbf{D}$ with additional forward and inverse access to the underlying primitives p, p^{-1}. The proof borrows ideas from APE's original privacy and authenticity proof [2]. Particularly, we again perform a PRP-PRF switch to replace the permutation by random functions (f, f^{-1}), which provide a random answer from $\mathbf{R}\times\mathbf{C}$ to every new query, and abort if this leads to a collision. We find:

$$\mathsf{PA1}^{\mathbf{E}}_{\Pi}(\mathbf{D}) = \underset{\mathbf{D}}{\Delta}(\mathcal{E}_K, \mathcal{D}_K, p, p^{-1}\,;\, \mathcal{E}_K, \mathbf{E}, p, p^{-1}) \le \frac{\sigma^2}{2^{r+c}} + \underset{\mathbf{D}}{\Delta}(\mathcal{E}_K, \mathcal{D}_K, f, f^{-1}\,;\, \mathcal{E}_K, \mathbf{E}, f, f^{-1}) \enspace ,$$

where we recall that $\mathbf{D}$ makes queries whose lengths in number of blocks sum up to σ.

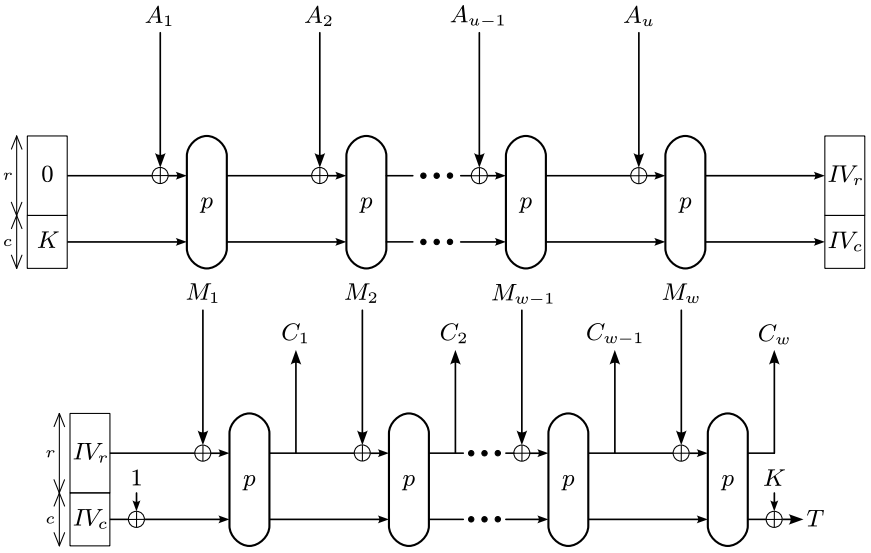

Fig. 4. APE: processing of associated data (top) and APE encryption mode (bottom). Here, p is an $(r+c)$-bit permutation. If $A = \varnothing$, set $IV = 0^r K$.

If f is called by $\mathbf{D}$ then we call this a direct f-query, and similar for direct f^{-1}-queries. A call of f by $\mathcal{E}_K$ or $\mathcal{D}_K$ (as a result of $\mathbf{D}$ calling them) is called an indirect f-query, and similar for indirect f^{-1}-queries (via $\mathcal{D}_K$). Every indirect f-query has a sequence of associated data blocks and/or message blocks leading up to it (from the $\mathcal{E}_K$- or $\mathcal{D}_K$-query calling it); we call this sequence the message chain associated to the indirect f-query. Likewise, every indirect f^{-1}-query has a tag and a sequence of ciphertext blocks leading to it, and we refer to it as the ciphertext chain. Let $\mathcal{Q}_i$ denote the set of all prefixes of all queries made by $\mathbf{D}$ to its $\mathcal{E}_K$-oracle before the ith (f, f^{-1})-query, where an $\mathcal{E}_K$-query (A, M) results in prefixes $\{A_1, A_1A_2, \ldots, AM\}$. In this set, we also include $\{A_1, \ldots, A\}$ for a $\mathcal{D}_K$-query (A, C, T) (queries to $\mathbf{E}$ do not add to $\mathcal{Q}_i$). Let $\mathcal{Q}_i^{-1}$ denote the set of all suffixes of all queries made by $\mathbf{D}$ to its $\mathcal{E}_K$- or $\mathcal{D}_K/\mathbf{E}$-oracle before the ith query, where a query tuple (A, M, C, T) results in suffixes $\{C_\ell T, C_{\ell-1}C_\ell T, \ldots, CT\}$. Regarding all *direct* queries before the ith query, we denote by X_i^{dir} the set of all capacity values input to f-queries or output of f^{-1}-queries, and by Y_i^{dir} the set of all capacity values input to f^{-1}-queries or output of f-queries. For example, a direct forward query $f(x) \to y$ adds $[x]_c$ to X_i^{dir} and $[y]_c$ to Y_i^{dir}, and a direct inverse query $f^{-1}(y) \to x$ adds $[x]_c$ to X_i^{dir} and $[y]_c$ to Y_i^{dir}. The sets X_i^{ind} and Y_i^{ind} are defined similarly. We write $X_i = X_i^{\text{dir}} \cup X_i^{\text{ind}}$ and $Y_i = Y_i^{\text{dir}} \cup Y_i^{\text{ind}}$, and initialize $X_0^{\text{ind}} = Y_0^{\text{ind}} = \{K\}$.

For this ith query to f, we define the following auxiliary events:

$$\begin{aligned}
\mathsf{E}_i^{\text{dir-}X} &: \text{ direct } f(x) \to y \text{ or } f^{-1}(y) \to x \text{ satisfies } [x]_c \in X_i^{\text{ind}} \cup X_i^{\text{ind}} \oplus 1 \ , \\
\mathsf{E}_i^{\text{ind-}X} &: \text{ indirect } f(x) \text{ with message chain } (A, M) \notin \mathcal{Q}_i \text{ satisfies} \\
&\qquad [f(x)]_c \in X_i \cup X_i \oplus 1 \ , \\
\mathsf{E}_i^{\text{dir-}Y} &: \text{ direct } f(x) \to y \text{ or } f^{-1}(y) \to x \text{ satisfies } [y]_c \in Y_i^{\text{ind}} \cup Y_i^{\text{ind}} \oplus 1 \ , \\
\mathsf{E}_i^{\text{ind-}Y} &: \text{ indirect } f^{-1}(y) \text{ with ciphertext chain } (C, T) \notin \mathcal{Q}_i^{-1} \text{ satisfies} \\
&\qquad [f^{-1}(y)]_c \in Y_i \cup Y_i \oplus 1 \text{ or } [y]_c \in Y_i^{\text{dir}} \oplus K \ .
\end{aligned}$$

We set $\mathsf{E}_i = \mathsf{E}_i^{\text{dir-}X} \cup \mathsf{E}_i^{\text{ind-}X} \cup \mathsf{E}_i^{\text{dir-}Y} \cup \mathsf{E}_i^{\text{ind-}Y}$, and furthermore define

$$\hat{\mathsf{E}}_i := \mathsf{E}_i \cap \textstyle\bigcap_{j=1}^{i-1} \overline{\mathsf{E}_j}, \text{ and } \mathsf{E} := \textstyle\bigcup_{i=1}^{\sigma} \hat{\mathsf{E}}_i \ ,$$

where $\overline{\mathsf{E}_j}$ is the complement of E_j. In Lem. 1, we prove that $(\mathcal{E}_K, \mathcal{D}_K, f, f^{-1})$ and $(\mathcal{E}_K, \mathbf{E}, f, f^{-1})$ are indistinguishable as long as E does not occur. Following the analysis of [2], in the

real world we have $\Pr[\mathbf{D}^{\mathcal{E}_K,\mathcal{D}_K,f,f^{-1}} \text{ sets } \mathsf{E}] \leq \frac{2\sigma(\sigma+1)}{2^c}$. The proof is now completed via the fundamental lemma of game playing [14].

Lemma 1. *Given that* E *does not occur,* $(\mathcal{E}_K, \mathcal{D}_K, f, f^{-1})$ *and* $(\mathcal{E}_K, \mathbf{E}, f, f^{-1})$ *are indistinguishable.*

Proof. Clearly, direct queries to f or f^{-1}, or indirect queries to f via $\mathcal{E}_K$, appear indistinguishable to $\mathbf{D}$ as long as they do not coincide with any queries coming from $\mathcal{D}_K$. This, indeed, never happens due to $\overline{\mathsf{E}^{\text{dir-}X}}$, $\overline{\mathsf{E}^{\text{dir-}Y}}$, and $\overline{\mathsf{E}^{\text{ind-}X}}$, respectively. It suffices to focus on decryption queries (to $\mathcal{D}_K$ or $\mathbf{E}$). Let (A, C, T) be a query made by $\mathbf{D}$. Denote by ℓ the number of blocks of C.

Firstly, consider the case $(C_j \cdots C_\ell, T) \in \mathcal{Q}_i^{-1}$ for some $j \in \{1, \ldots, \ell+1\}$ and assume j is minimal (we will come back to the case of $(*, T) \notin \mathcal{Q}_i^{-1}$ later in the proof). Let (A', M', C', T') be the corresponding earlier query and denote its block length by ℓ', so $C_j \cdots C_\ell = C'_{\ell'-\ell+j} \cdots C'_{\ell'}$. This tuple could have been defined via an encryption query (hence the adversary queried (A', M') to $\mathcal{E}_K$) or via a decryption query (hence the adversary queried (A', C', T') to $\mathcal{D}_K/\mathbf{E}$).

By construction, if $\mathbf{D}$ is conversing with $\mathbf{E}$, we have $M_i = M'_{\ell'-\ell+i}$ for $i = \ell, \ldots, j+1$, $M_j = M'_{\ell'-\ell+j} \oplus C'_{\ell'-\ell+j-1} \oplus C_{j-1}$, and M_i uniformly at random for $i = 1, \ldots, j-1$. We will consider how these values are distributed for the real decryption algorithm $\mathcal{D}_K$. The values M_i for $i = \ell, \ldots, j+1$ are as before (this follows clearly from the specification of $\mathcal{D}_K$), and we focus on the remaining values. We distinguish between $j > 1$ and $j = 1$.

- $j > 1$. Write the indirect query corresponding to the computation of M_j as $f^{-1}(y)$. By construction, $f^{-1}(y)$ is no new query: it has been queried in the evaluation of (A', M', C', T'), and particularly satisfies $[f^{-1}(y)]_c = M'_{\ell'-\ell+j} \oplus C'_{\ell'-\ell+j-1}$. In the current case, $\mathcal{D}_K$ sets $M_j = [f^{-1}(y)]_c \oplus C_{j-1}$ by construction, hence following the same distribution as $\mathbf{E}$.
  The next query in the evaluation, write $f^{-1}(y')$ where $y' = f^{-1}(y) \oplus C_{j-1} \| 0^r$, is new. Indeed, suppose it was already made before. This could have happened in an indirect or direct query. For the former case, then $[f^{-1}(y)]_c$, as range value input to f^{-1}, is in Y^{dir}. This would imply that the query $f^{-1}(y)$ once hit a direct query (impossible by $\overline{\mathsf{E}^{\text{ind-}Y}}$) or vice versa (impossible by $\overline{\mathsf{E}^{\text{dir-}Y}}$). For the latter case, by $\overline{\mathsf{E}^{\text{ind-}X}}$ and $\overline{\mathsf{E}^{\text{ind-}Y}}$, this would imply $(C_{j-1} \cdots C_\ell, T) \in \mathcal{Q}_i^{-1}$, contradicting minimality of j. Consequently, the response and thus also M_{j-1} are randomly drawn, and so forth until M_1. Here, we rely that the last indirect query never matches a direct query $\oplus K$ (by $\overline{\mathsf{E}_i^{\text{ind-}Y}}$).
- $j = 1$. Denote the state coming from $\mathsf{hash\text{-}data}_{0,K}(A)$ by IV (if $A = \varnothing$, $IV = (0, K)$). Essentially the same reasoning implies that $M_1 = M'_{\ell'-\ell+1} \oplus C'_{\ell'-\ell} \oplus IV_r$ (note that $\ell' \geq \ell$ as $j = 1$).

Finally, if $(*, T) \notin \mathcal{Q}_i$, hence this is the first time a query for this particular tag T is made, the above reasoning carries over for $j = \ell$. □

□