Rogue Decryption Failures: Reconciling AE Robustness Notions

Guy Barwell, Daniel Page, and Martijn Stam

Department of Computer Science, University of Bristol,
Merchant Venturers Building, Woodland Road,
Bristol, BS8 1UB, United Kingdom.
{guy.barwell, daniel.page, martijn.stam}@bristol.ac.uk

Abstract. An Authenticated Encryption scheme (AE) is deemed secure if ciphertexts both look like random bitstrings and are unforgeable. One shortcoming of AE as commonly understood is its idealized, all-or-nothing decryption: if decryption fails, it will always provide the *same single* error message *and nothing more*. Reality often turns out differently: encode-then-encipher schemes often output decrypted ciphertext before verification has taken place, whereas pad-then-MAC-then-encrypt schemes are prone to distinguishable verification failures due to the subtle interaction between padding and the MAC-then-encrypt concept. Three recent papers provided what appeared independent and radically different definitions to model this type of decryption leakage.
To reconcile these three works, and indeed the literature in general, we define an expressive "clean slate" framework that allows us to compare and contrast the previous notions within a systematic naming scheme. We then extend this by allowing for (deterministic) decryption leakage from invalid queries, providing a reference model of security we term Subtle Authenticated Encryption (SAE). Then, we thoroughly describe this landscape by translating classical results (where applicable) and extending them to encompass our new notions. Finally, with SAE as a reference point, we compare the three noted works. We find that, at their core, the previous notions are essentially equivalent: their key differences stem from definitional choices independent of the desire to capture real world behaviour.

Keywords: provable security, authenticated encryption, multiple errors, unverified plaintext, robustness

An earlier version of this work was presented at IMA Coding and Cryptography 2015 [6] and published by Springer International Publishing, DOI:10.1007/978-3-319-27239-9_6. This is the full version, released Saturday 7th May, 2016.

Table of Contents

1 Introduction

Nowadays, Authenticated Encryption (AE) is understood to mean that ciphertexts both look like random bitstrings (which we term IND–CPA, following recent literature [3, 31]) and are unforgeable (INT–CTXT). Moreover, the customary syntax of AE considers encryption as deterministic and stateless, instead accepting a nonce (number-used-once) and associated data to ensure that repeated encryption of the same message does not lead to repeated ciphertexts. Security is expected to hold as long as each nonce is used at most once for encryption, and preferably security degrades gracefully even when nonces are repeated. AE, thus defined, is more flexible and considerably stronger than the traditional notion of probabilistic, left-or-right indistinguishable symmetric encryption under chosen ciphertext attacks (LOR–CCA, or IND–CCA2 in older literature [10]). Indeed, for length-regular schemes and with nonces chosen uniformly from a sufficiently large domain, IND–CPA and INT–CTXT together imply textbook LOR–CCA security.

The ongoing CAESAR competition [16] has served as a catalyst to strengthen the security models used in AE even further. One particular shortcoming is the traditional reliance on an idealised, all-or-nothing decryption: if decryption fails, it will only ever provide a single error message. For various reasons, this is not a realistic assumption. Especially MAC-then-encrypt schemes (or rather, decrypt-then-verify) are prone to real-world security flaws, on the one hand due to distinguishable verification failures and on the other due to the need to output (or at least store) decrypted ciphertext before verification has taken place.

Three recent works improve the "robustness" of AE schemes by considering how well their security guarantees hold up under incorrect usage or when implemented non-ideally.[1] Boldyreva et al. [19] investigated the effect of multiple decryption errors for both probabilistic and stateful encryption (BDPS). Later, Andreeva et al. [3] moved to a nonce-based setting, introducing a framework to capture the Release of Unverified Plaintexts (RUP). Concurrently, Hoang et al. [29] coined an alternative notion, Robust Authenticated Encryption (RAE), which they claimed radically different from RUP.

On the surface, these papers take very different approaches, with quite different goals in mind. Their decryption functions all have a different syntax, and they all use different methods to characterise valid or invalid ciphertexts. BDPS concentrates on decryption errors and does not consider nonce-based encryption. RUP extends AE by syntactically adding explicit, fixed-size tags and considering separate verification and decryption algorithms. It models the leakage of candidate plaintexts, with an eye on both the online and nonce-abuse settings. Finally, RAE considers schemes with variable, user-specified stretch as authentication mechanism, and decryption is given a much richer syntax, extending semantics for ciphertexts not generated by the encryption algorithm. This raises the natural questions how these models relate to each other and how well each captures real-world decryption leakage.

Our contribution. Inspired by the above works, we provide a framework combining the best of all worlds, where our overarching goal is to bring together all the recent developments towards a strengthened AE notion within a single unified formalisation. In doing so, we are able to reconcile RUP and RAE with BDPS, both notationally and conceptually. Our framework allows us to draw parallels and highlight where the works agree or differ, while ensuring any goals described can be easily interpreted and compared to the scenarios they purport to model.

We begin by defining a broad reference game that models adversarial access, and provide a systematic naming scheme for all the associated security notions. Within this, we define "Subtle Authenticated Encryption" (SAE) as the strongest security goal relevant to (deterministic) decryption leakage. The term *subtle* highlights that security in the real-world is very much dependent of the subtleties of *how* decryption is implemented. As illustration, we describe a natural yet insecure implementation of AEZ [29], refuting its robustness.

[1] The term "implemented non-ideally" is not meant to suggest any fault on behalf of the implementer, but to acknowledge that, in the real world, "ideal" implementations are not always feasible.

With this reference game in place, we provide a comprehensive study of the implications and separations between the notions, including (reassuringly) recovering the classical IND–CPA + INT–CTXT composition result, and observing that IND–CCA with passive integrity also achieves AE. We discuss the concept of simulatable leakage and introduce the goal of "error simulatability" to formally measure of how significant leakage is, allowing us to generalise this decomposition into the subtle case. Finally, we compare results from the three noted papers, using SAE as a reference point. After clarifying some (misconceived) terminology, we find that for schemes *with fixed stretch* the notions essentially coincide.

The fundamental difference between the models is philosophical: Is Authenticated Encryption primarily a primitive like a blockcipher, whose security should be measured with reference to the ideal object of the given syntax and where the authentication level might be set to zero, corresponding to RAE's variable stretch—expecting sufficiently savvy users to use the "right" stretch for their purposes; or is it a means to authenticate and encrypt where security should be measured against a—possibly unobtainable—ideal, but with little room for misunderstanding by users?

2 Security Games for the Real World

2.1 Standard Syntax of Authenticated Encryption

Current understanding of authenticated encryption is the culmination of many years of work (see Appendix A for an overview). Modern AEAD schemes take a number of standard inputs and produce a single output. The corresponding spaces are named after the elements they represent: the *key* space K, the *message* space M, the *nonce* space N, the *associated data* space A, and finally the *ciphertext* space C. Each of these spaces is a subset of $\{0,1\}^*$, but we make no assertions over their sizes.

An *authenticated encryption* (AE) scheme is a pair of deterministic algorithms, $\mathcal{E}$ for encryption and $\mathcal{D}$) for decryption, satisfying

$$\begin{aligned}\mathcal{E} &: \mathsf{K} \times \mathsf{N} \times \mathsf{A} \times \mathsf{M} \to \mathsf{C} \\ \mathcal{D} &: \mathsf{K} \times \mathsf{N} \times \mathsf{A} \times \mathsf{C} \to \mathsf{M} \cup \{\perp\}\ .\end{aligned}$$

We use subscripts for keys, superscripts for public information (nonce and associated data) and put content data in parentheses.

To be *correct*, decryption must be a left inverse of encryption: if $C = \mathcal{E}_k^{N,A}(M)$ then $\mathcal{D}_k^{N,A}(C) = M$. Conversely, a scheme is *tidy* if decryption is a right inverse: if $\mathcal{D}_k^{N,A}(C) = M \neq \perp$ then $\mathcal{E}_k^{N,A}(M) = C$. Together then, correctness and tidiness imply encryption and decryption are inverses. For schemes that are both correct and tidy, $\mathcal{E}_k$ uniquely determines $\mathcal{D}_k$, which implies that security can be regarded as a property of $\mathcal{E}_k$ only [42].

The *stretch* measures the amount of ciphertext expansion (or redundancy). We require that the stretch $\tau(M) = |\mathcal{E}_k^{N,A}(M)| - |M|$ depends only on the length of the message (for all k, N, A and M). We call such schemes *τ-length-regular*, extending the accepted term *length-regular* to describe *how* the length is regulated, and henceforth restrict ourselves to length-regular schemes. Almost all modern schemes *are* length-regular and a significant portion even have constant stretch τ (to minimise ciphertext expansion). The syntax above is standard, but deviations exist. On the one hand, RUP uses an equivalent formulation with explicit tag space in addition to the ciphertext space (see Section 3.2). On the other hand, RAE uses an explicit input of the encryption indicating what size of tag is desired. In Section 3.3 we discuss the implication of user-defined tag-sizes and why we omit these from our framework.

2.2 Syntax of Subtle Authenticated Encryption

Just as a plan seldom survives contact with the enemy, so it goes with authenticated encryption: several provably secure schemes have fallen when implemented in practice. Especially for the decryption of invalid ciphertexts, it is challenging to ensure that an adversary *really* only learns the invalidity of the ciphertext, and not some additional information. Additional information that has been considered in the

past (and we will encounter again shortly) are multiple error symbols and unverified plaintext. Both can be classified as *leakage*, leading to our new notion of a *subtle authenticated encryption* (SAE) scheme.

An SAE scheme is a pair of deterministic algorithms $(\mathcal{E}, \Lambda)$, where Λ corresponds to leakage from the decryption function. Closely associated to it is the decryption function $\mathcal{D}$ induced by $\mathcal{E}$, and so at times we will consider the scheme as a triple $(\mathcal{E}, \mathcal{D}, \Lambda)$. We restrict ourselves to leakage functions that are deterministic functions on their inputs and only provide leakage to invalid decryption queries. Thus the leakage function

$$\Lambda : \mathsf{K} \times \mathsf{N} \times \mathsf{A} \times \mathsf{C} \to \{\top\} \cup \mathsf{L}$$

is such that the *leakage space* L can be any non-empty set not containing $\top$, and the distinguished symbol $\top$ refers to a message that is valid. So, for any (N, A, C), either $\mathcal{D}_k^{N,A}(C) = \perp$ or $\Lambda_k^{N,A}(C) = \top$, but not both: a message is either valid (and so decryption returns the plaintext but there is no leakage) or is invalid (and so decrypts to $\perp$ and leakage is available). The generality of L caters for any type of leakage when presented invalid ciphertexts, including schemes with multiple errors [19], those which output candidate plaintexts [3] or return arbitrary strings [29], or indeed the classical case where nothing is leaked.

Explicitly separating Λ from $\mathcal{D}$ emphasises that leakage is a property of the decryption *implementation*, rather than of the decryption *function*. Consequently, security (for correct and tidy schemes) becomes a property of both the encryption function *and* the decryption implementation's leakage. A scheme may be proven secure for some leakage model Λ, but such a result is only meaningful as long as Λ accurately reflects the actual leakage as observed in practice. Even minor optimizations of the same decryption function can change the associated implementation so much that the scheme goes from being provably secure under some robust security definition to trivially insecure. We show how AEZ is affected by implementation choices in Appendix E. From this perspective, security becomes a subtle rather than robust affair, hence the name *subtle authenticated encryption*, inspired by the *SubtleCrypto* interface of the WebCryptoApi [53].

Comparison with the traditional model. There is a canonical mapping from any SAE scheme $(\mathcal{E}, \mathcal{D}, \Lambda)$ to a more traditional one $(\mathcal{E}, \mathcal{D})$ simply by removing access to the leakage oracle: correctness and tidiness of the subtle scheme clearly imply correctness and tidiness of the traditional one. Note that many distinct SAE schemes map to the same traditional form, implying that the canonical mapping induces an equivalence relation on SAE schemes. One could turn a traditional scheme $(\mathcal{E}, \mathcal{D})$ into a subtle form by inverting the above canonical map, for which the obvious preimage is setting $\Lambda^{N,A}(C) = \perp$ if $\mathcal{D}_k^{N,A}(C) = \perp$ and otherwise $\top$, again preserving correctness and tidiness. This corresponds to the SAE scheme whose implementation does not leak at all, so we expect our security notion to match the traditional one in this case (and it does).

Contrast with leakage resilience. Our separation into $\mathcal{D}$ and Λ is possible because decryption is deterministic, stateless, and its inputs, the triple (N, A, C), may be provided to Λ_k. Within the leakage resilience community [25], leakage is generally characterised as an auxiliary output from the original algorithm (often supported by an auxiliary input to control the type of leakage); moreover one would expect *both* encryption and decryption to leak. This integrated perspective reflects the real world more closely (as leakage results from running some algorithm) and is more expressive. For example, if the decryption function were probabilistic, the leakage may require access to the internal randomness, or if the scheme is stateful it may require the correct state variables. Some of these issues could be overcome by (for example) assuming the adversary always calls $\mathcal{D}_k$ directly before calling Λ_k, and that Λ_k has access to the previous internal state (from which it can deduce the operation of $\mathcal{D}_k$ if required).

Ultimately which syntax works best depends on the context. We feel that, in the context of capturing subtle implementation differences for modern authenticated encryption (where decryption is stateless and deterministic), separating leakage and decryption is a useful abstraction. Our work could be recast into a form more closely aligned with the leakage resilience literature, but the notation would become

more cumbersome, especially when an adversary can only observe the leakage (corresponding to the sPAS scenario in the next section).

2.3 Authentication and Encryption Security Games

In most modern AE definitions, an adversary is given access to a pair of oracles claiming to implement encryption and decryption. They are either real, and act as claimed, or ideal, returning the appropriate number of random bits for encryptions and rejecting all decryption attempts. To win the game, the adversary must decide which version it is interacting with. Certain queries would lead to trivial wins, for example asking for the decryption of a message output by the encryption oracle. These queries are forbidden (or their output suppressed).

This contrasts with the original definition of AE as LOR–CPA + INT–CTXT, where in both constituent games an adversary only has access to a single, real encryption oracle (and no decryption oracle); moreover, classically in the LOR–CPA game only a single challenge ciphertext is present and for INT–CTXT only a single ciphertext needs to be forged. At first sight the two definitions may appear quite different, yet they are known to be equivalent. Where does this difference stem from and should one be preferred over the other?

We argue that both definitions can be cast as simplifications of a single reference game. This reference game is itself a distinguishing game where an adversary has access to two sets of oracles: one set of oracles will be used to capture the *goal* of the adversary, whereas the other matches the *powers* of the adversary. For instance, to capture AE an adversary has access to *four* oracles: the two oracles from the modern definition (implementing either the real or ideal scenario) *and* the two oracles from the traditional LOR–CCA definition (namely true encryption and decryption oracles).

Starting from the reference game one can use a hybrid argument to show that, from a theoretical perspective, two types simplifications are possible. Either the true encryption and/or decryption oracles are superfluous; or restriction to a single query to each challenge oracle suffices. One cannot, however, make both simplifications simultaneously. It has become customary to only study some simplified game, yet we posit that our approach using a reference game (with up to five oracles) has several advantages:

1. *Generality:* Hybrid arguments and composition results—the techniques implicitly underlying the standard definition—do not always hold when enriching the security model to take into account real-world phenomena such as key dependent messages or leakage (e.g. [23]). In these cases, one typically undoes certain simplifications; relying on our reference game instead is more transparent.
2. *Granularity:* Because adversarial goal and power are clearly separated, one can immediately identify a natural lattice of security notions and argue about possible equivalences *depending on the context*.
3. *Intuition:* The simplified games are less intuitive when considering real-world scenarios. For instance, even if an adversary has seen a number of true plaintext–ciphertext pairs, for any set of fresh purported plaintext–ciphertext pairs it should be clueless as to its validity. This statement follows directly from our reference game, yet for the simplified games one would need a hybrid argument.
4. *Tightness:* In real world scenarios, obtaining challenge ciphertexts versus known ciphertexts might carry different costs, which can be more easily reflected in our reference game (as the queries go to different oracles). A security analysis directly in our game is potentially more tight than one in a simplified game (whose results subsequently need to be ported to the more fine-grained real-world setting).

Notation. We refer to games in the form GOAL–POWER, clearly separating the adversary's objective from its resources. The complete lists of powers and goals are represented in Figure 1, and described below. Security of scheme $(\mathcal{E}, \Lambda)$ in game X against an adversary $\mathcal{A}$ is written as an advantage $\mathsf{Adv}^{\mathrm{X}}_{\mathcal{E},\Lambda}(\mathcal{A})$ and captures the adversary's ability to distinguish between two worlds. In both worlds the adversary has oracle access that depends on the scheme (initiated using some random and secret key $k \leftarrow\!\!\$\ \mathsf{K}$); the oracles corresponding to the goal differ between the worlds, whereas the oracles corresponding to the power will be identical. Without loss of generality, oracles will not be called on elements outside of

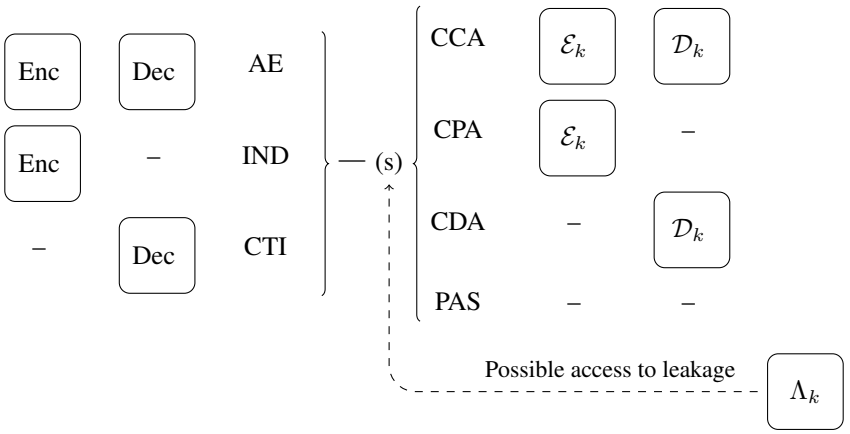

Fig. 1: A representation of the $24 = 3 \cdot 4 \cdot 2$ games. The challenge oracles Enc and Dec specify the adversary's goal: they either implement honest encryption and decryption or their idealised versions, where Enc samples responses randomly and Dec returns $\perp$ to all queries. The honest oracles $\mathcal{E}_k$ and $\mathcal{D}_k$ capture the adversary's power. Each game corresponds to a 5-bit bitstring $\mathtt{vwxyz}$, with for example CTI–CPA (equivalent to INT–CTXT) being 01100, and IND–sCCA (i.e. IND–CCA with decryption leakage) as 10111.

the appropriate spaces. A scheme is deemed X *secure* if $\mathsf{Adv}^{\mathrm{X}}_{\mathcal{E},\Lambda}$ is sufficiently small for all reasonably resourced adversaries.

Goals. The goal oracles Enc and Dec either implement the true scheme or an idealised version. If $b = 0$, we are in the *real world*, where Enc and Dec implement $\mathcal{E}_k$ and $\mathcal{D}_k$ respectively, whereas if $b = 1$ we are in the *ideal world*, where they implement $\$$ and $\perp$, respectively.

The oracle $\perp$ matches the syntax of $\mathcal{D}_k$ but returns $\perp$ in response to any queries. The oracle $\$$ is a random function: for each nonce–associated-data pair, it samples an element $\$_{N,A}$ uniformly at random from the set of all τ-length-regular functions $f : \mathsf{M} \to \mathsf{C}$.[2] When queried, $\$(N, A, M) := \$_{N,A}(M)$. Ignoring repeated queries, this corresponds to uniformly and freshly sampling $|M| + \tau(|M|)$ random bits for each query.

The *goal* is defined based on which oracles an adversary is given access to. We code this access using 2-bit strings ($\mathtt{vw}$), where the first bit is set in the presence of an Enc oracle, leaving the second bit for Dec. This leads to *three* possible goals (it does not make sense to have no challenge oracle): indistinguishability (IND, 10); authenticated encryption (AE, 11); and ciphertext integrity (CTI, 01).

Two small notes on nomenclature are in order. Firstly, indistinguishability from random ciphertexts was originally introduced [48] as IND$, with IND itself used to denote LOR (left or right) security. Since the LOR game is hardly ever used in modern literature, like others before us (e.g. [3]) we have repurposed IND to denote what used to be IND$. We are less enthused about the use of the term "priv" (shorthand for privacy) to denote what was IND$–CPA (see e.g. [31,32,37,40]) as it neither captures the attack–goal concept, nor does it acknowledge that privacy extends beyond data confidentiality. Whereas confidentiality protects just the *content* of a message, privacy ought to hide the *context* in which a message is sent (often referred to as "metadata") as well [56].

Secondly, we use the abbreviation CTI for ciphertext integrity as a goal (so we can still attach a power to it), as opposed to the commonly encountered INT–CTXT as shorthand for ciphertext integrity incorporating both goal and power.

[2] Formally, the $\$_{N,A}$ are defined adaptively, to avoid the issue of uniformly sampling from an infinite set. whenever they are queried with an input (N, A, M) such that N, A or $m = |M|$ is new, the injection $\$_{N,A,m}$ is sampled uniformly from all functions from m to $\tau(m)$ bits. The function then returns $\$(N, A, M) := \$_{N,A,m}(M)$.

Ideal versus attainable. Our ideal encryption oracle responds with random bitstrings for fresh calls. This corresponds to security as one would expect it to hold; it can be considered as a computational analogue of Shannon's notion of perfect security, where the uncertainty of a ciphertext given a message should be maximal. Similarly, the ideal decryption oracle is unforgiving, implying (traditional) integrity of ciphertexts.

Consequently, for some classes of constructions the advantage cannot be small. For instance, when nonces are repeated in so-called online schemes, it will be easy to distinguish real encryption from random by looking at prefixes. One can bypass these impossibilities by adapting the ideal oracles accordingly [2,30]. As another example, for schemes with insufficient stretch, a randomly chosen bitstring is likely a valid ciphertext (leading to a forgery). Hoang et al. [29] suggest to use attainable security as the benchmark instead, see Section 3.3 for further discussion.

Powers. Traditionally, an adversary's *powers* describe what access they are given to honest encryption and decryption oracles, with which to learn about the scheme. Again, we identify these with 2-bit strings $\mathtt{xy}$, visualised in Figure 1. The standard notions are a *passive* attack (PAS, 00), a *chosen plaintext* attack (CPA, 10), and a *chosen ciphertext* attack (CCA, 11). Access to only a decryption oracle is known as (DEM) CCA in the KEM–DEM setting (e.g. [21,22]), we will refer to it as a *chosen decryption* attack (CDA, 01).[3] Unless *overall* encryption access is restricted as in the DEM scenario, the CDA scenario is of limited relevance (see Section 2.4).

The leakage oracle. We add a third honest oracle implementing Λ_k, that models how schemes behave when subject to imperfect decryption implementations. Again, we use a bit ($\mathtt{z}$) to indicate whether a game provides an adversary access or not, appending it to the string of power oracles. If not, the standard notions arise, but presence leads to a range of new notions, which we will call the "subtle" variants. The name is chosen to emphasise that security critically depends on implementation subtleties.

As an example, power 101 stands for "subtle Chosen Plaintext Attack", or sCPA in short (note the "s" prefix). The power 001 corresponds to an adversary who cannot make decryption queries, yet it can observe leakage from them. This seeming contradiction makes sense when recalling that Λ_k yields information when queried with invalid ciphertexts. For instance, an adversary might be able to learn how long it takes for ciphertexts to be rejected, but not what plaintexts correspond to valid ciphertexts. Given the implied validity checking of this notion, it has been referred to as a *Ciphertext Validity Attack* [7,19] (CVA). In favour of internal consistency, we term it the sPAS attack, which can be thought of as passively observing leakage.

The role of nonces. The more control an adversary has over the nonces when it makes a query, the stronger the security notion. Usually an adversary has full control over the nonces used by decryption (corresponding to the traditional view where the nonce is considered *part of* the ciphertext), but for encryption there are various degrees. Rogaway and Shrimpton [51] and later Namprempre et al. [42] describe three options: for IV-based schemes, the game randomly selects a fresh nonce for each encryption query; for nonce-based encryption, the adversary selects the nonce, but respects unicity (so indeed only uses each nonce once); finally for deterministic encryption, the adversary has full control over the nonce. An adversary is termed *nonce-respecting* if he does not query (N, A', M') to either Enc or $\mathcal{E}_k$ if he has already queried either of them with (N, A, M) for some A and M.

These notions can each be formalized either in terms of adversarial behaviour or, alternatively, enforced by the game (e.g. by returning $\lightning$ to suppress encryption queries that repeat a nonce). Suppressing queries potentially gives a stronger (and more meaningful) security notion, for instance in a stateful setting, where an invalid query might still advance the state. However, in the nonce-based context we consider the formalisations are equivalent and defining security by ranging over the appropriate class of adversaries is—in our opinion—easier and more intuitive to deal with.

[3] In the context of public key encryption, CDA has been used for chosen distribution attacks [9], which is a completely different concept (and irrelevant when considering nonce-based symmetric primitives).

Consequently, when defining a security notion, we will provide a single game independent of how nonces are treated, which subsequently leads to two flavours of the notion depending on the type of adversaries allowed: if it is nonce-respecting we get nonce-based security and, finally, if the adversary is potentially nonce-abusing then deterministic encryption (often called "misuse resistant") emerges. Henceforth we will concentrate on the nonce-respecting scenario, as for any reduction one has to show it preserves nonce-respecting behaviour. Since nonce-abusing behaviour is automatically preserved, all our results apply equally to the more desirable deterministic or misuse-resistant scenario.

At this point, it would be tempting to assert that our work will hold in the IV-based scenario as well. If the adversary is *nonce-randomizing* (i.e. samples a uniform random 'nonce' just before sending an encryption query) one might expect to obtain IV-based encryption, except that it does not quite. In the next paragraph we will introduce prohibited and pointless queries and we'll see that the more control an adversary has over the input the more restrictions are required to avoid trivial wins. In particular, in the IV-based case, an adversary has no control over nonces used during encryption queries, whereas an adversary gains control when moving to a nonce-based setting. The result is that certain encryption queries that are acceptable when considering random IVs, become prohibited in the nonce-based setting.

We will address the surprising consequences of this difference between IV-based and nonce-based security in Appendix B. In many cases, one can still apply our results to the IV-based scenario at the cost of an additional term of $\sim q^2/|N|$ to bound the possibility of collisions on the IV space.

'Prohibited' and pointless queries. With unfettered oracle access, certain (sequences of) queries would lead to easily distinguishable behaviour between the real and ideal worlds, giving rise to trivial wins. For instance, sending a true ciphertext (obtained from $\mathcal{E}_k$) to Dec will always output $\perp$ in the ideal world, yet never in the real one. We call this *forwarding* a query from $\mathcal{E}_k$ to $\mathcal{D}_k$, and, to avoid trivial wins, we must somehow prevent the adversary from utilizing this. We do so by letting the game artificially suppresses the output of the (in this case) problematic Dec query, outputting $\lightning$ instead. For reasons that will become apparent shortly, we call these queries *prohibited*. On the other hand, some queries are harmless but *pointless* for an adversary to make. For instance, repeating a decryption query will simply give the same result again.

In the reference game, the adversary may make any queries; it is the game's responsibility to suppress outputs to prevent trivial wins. However, in our games the adversary can always tell in advance whether a query will be suppressed, moreover the query does not change the state of the game. Hence, without loss of generality, rather than suppressing the query's output we could *prohibit* the adversary from making it in the first place. We stress that when adapting our framework in other contexts (e.g. for leakage resilience) one should initially suppress outputs and ensure that any simplification to the effective game does not limit the adversary (cf. [12]).

When adversaries refrain from making prohibited and pointless queries, we arrive at the *effective* security game. Figure 2 gives an overview of both prohibited and pointless queries as explained below.

Prohibited queries. Any inputs (N, A, M) repeated between the two encryption oracles will distinguish the Enc oracle. Similarly, attempting to decrypt the output of Enc will allow the adversary to immediately determine whether Enc is random. Attempting to decrypt the output of the honest encryption oracle $\mathcal{E}_k$ will also trivially identify whether Dec is real or idealised. We also prohibit the adversary from forwarding any output from their $\mathcal{D}_k$ oracle to the encryption challenge oracle Enc, because the tidiness property implies that in the real world $\mathrm{Enc}(\mathcal{D}_k(C)) = C$, yielding a trivial distinguisher. This final restriction is commonplace in the nonce-based paradigm, following key previous papers such as the original nonce-based IND–CCA definition of Rogaway [48, §6] (there termed IND$–CCA), but is not made in the probabilistic world. The effect of this discrepancy is discussed further in Appendix B.

The same prohibitions are in place for the inputs to Λ_k as for $\mathcal{D}_k$; however we do not place any restrictions on the output of Λ_k. Any seemingly trivial wins that might occur from forwarding Λ_k's output should be considered a weakness of (the implementation of) the scheme and demonstrate that this particular Λ_k cannot be secure.

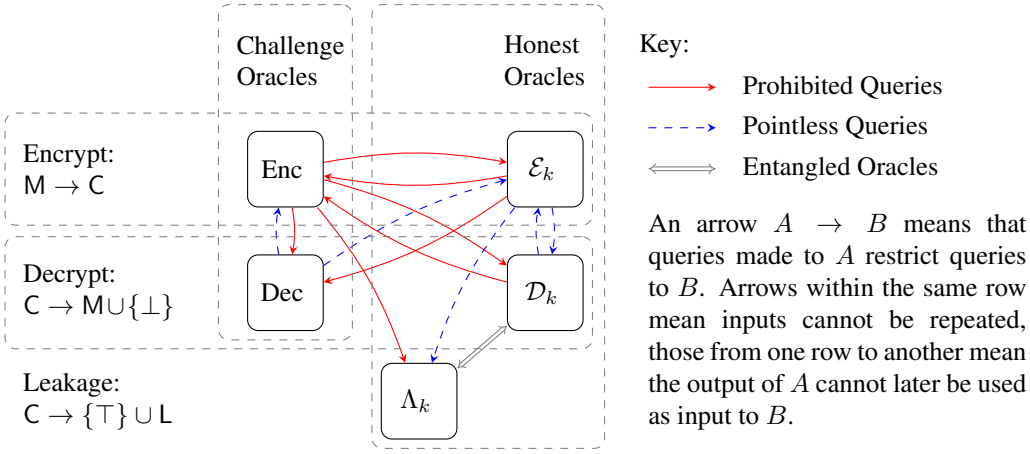

Fig. 2: Prohibited queries are those that allow the adversary trivial wins, and so must be suppressed. Pointless queries are those that cannot help an adversary win the game. In the reference game, the output of prohibited queries is suppressed; in the effective game, adversaries refrain from making either prohibited or pointless queries. The oracles $\mathcal{D}_k$ and Λ_k are entangled as they both correspond to (different aspects of) the decryption algorithm and its implementation: it would not make sense to block queries between them.

Pointless queries. Sending the output of $\mathcal{E}_k$ to $\mathcal{D}_k$ is pointless since by correctness the answer is already known. Similarly, tidiness implies the opposite: output from $\mathcal{D}_k$ need not be sent to $\mathcal{E}_k$. As soon as Dec outputs something other than $\perp$, the adversary can distinguish it as the real case, and so might as well terminate: this means no outputs from Dec need ever be queried to the encryption oracle(s). Finally, since the game is deterministic and stateless there is no point in repeating queries (this restriction is not displayed in the diagram).

2.4 Equivalences and Separations

Since there are 24 possible games (3 goals times 8 possible powers), the obvious question is how they relate to each other. Certainly for the classical setting (i.e. in the absence of a leakage oracle) various relations are known already, but not necessarily using modern notation (e.g. results may have initially been proven for the case of probabilistic encryption [10]).

We provide a comprehensive overview, writing X $\Longrightarrow$ Y to signify that security in game X implies security in game Y, meaning that for any adversary $\mathcal{A}$ against game Y there is an adversary $\mathcal{B}$ against game X who uses similar resources and wins with similar probability.

Proposition 1 lists three (classes of) implications, which allows us to reduce the 24 games to only 10 interesting ones in Corollary 3 (5 games with or without leakage) . We present our results in terms of the corresponding bitstring, where `v`, `w`, `x`, `y`, and `z` signify bits that may (but need not) be set.

Proposition 1 (Effective Games). *We may assume the adversary is deterministic and makes no pointless or prohibited queries. Against such an adversary, several games are trivially related:*

1. *Adding extra oracles never makes the adversary weaker.*
2. *`1w0yz` $\Longleftrightarrow$ `1w1yz`: an encryption oracle does not help if an Enc challenge oracle is present.*
3. *`v1x0z` $\Longleftrightarrow$ `v1x1z`: a decryption oracle does not help if a Dec challenge oracle is present.*

Proof. Consider a probabilistic adversary $\mathcal{A}$ as being a deterministic Turing machine with access to some finite randomness tape. Since the advantage is averaged across all possible tapes, there exists some tape that is at least as successful as the average $\mathcal{A}$. By fixing to this particular random tape, we arrive at a deterministic adversary with advantage at least that of $\mathcal{A}$.

For any adversary making pointless or prohibited queries, there is an equivalent one who does not, since they already know what the answer will be. Henceforth, we assume adversaries are self-censoring and refer to such adversaries as *effective* (corresponding to our notion of effective games).

Adding extra oracles never weakens an adversary, since they may simply ignore the ones they are not interested in. This immediately takes care of the $\Rightarrow$ direction for points 2 and 3. We now show that, in these specific cases, the added oracles do not help either, namely that security in the case with fewer oracles implies security even with the additional oracles. Point 2 asserts that an honest $\mathcal{D}_k$ oracle is not helpful if a Dec challenge oracle is available, and Point 3 gives the corresponding result for $\mathcal{E}_k$ and Enc.

Let $\mathcal{A}$ be an adversary against the game $1\mathtt{w}1\mathtt{yz}$, then, mildly abusing notation, we can write

$$\mathsf{Adv}^{1\mathtt{w}1\mathtt{yz}}_{\mathcal{E},\Lambda}(\mathcal{A}) = \Delta^{\mathcal{E}_k,\mathcal{D}_k,\mathcal{E}_k,\mathcal{D}_k,\Lambda_k}_{\$,\perp,\mathcal{E}_k,\mathcal{D}_k,\Lambda_k}(\mathcal{A})$$

with the understanding that only the Enc and $\mathcal{E}_k$ oracles (corresponding to the first respectively third oracles in the sequences above) are necessarily available to the adversary, since the presence of the others depends on $\mathtt{wyz}$. The triangle inequality yields that:

$$\Delta^{\mathcal{E}_k,\mathcal{D}_k,\mathcal{E}_k,\mathcal{D}_k,\Lambda_k}_{\$,\perp,\mathcal{E}_k,\mathcal{D}_k,\Lambda_k}(\mathcal{A}) \le \overbrace{\Delta^{\mathcal{E}_k,\mathcal{D}_k,\mathcal{E}_k,\mathcal{D}_k,\Lambda_k}_{\$,\perp,\$,\mathcal{D}_k,\Lambda_k}(\mathcal{A})}^{\text{(I)}} + \overbrace{\Delta^{\$,\perp,\$,\mathcal{D}_k,\Lambda_k}_{\$,\perp,\mathcal{E}_k,\mathcal{D}_k,\Lambda_k}(\mathcal{A})}^{\text{(II)}}.$$

We claim that there are adversaries $\mathcal{B}$ and $\mathcal{C}$ against $\mathsf{Adv}^{1\mathtt{w}0\mathtt{yz}}_{\mathcal{E},\Lambda}$ such that $\mathcal{B}$'s advantage corresponds to the (I) term above and that of $\mathcal{C}$'s to (II). Moreover, the running time of both adversaries is comparable to that of $\mathcal{A}$ as they run $\mathcal{A}$ as black box and simulate its environment by appropriate calling and forwaring to their own oracles. Specifically, $\mathcal{B}$ answers $\mathcal{A}$'s $\mathcal{E}_k$ queries using his own Enc oracle and forwarding all other queries, whereas $\mathcal{C}$ answers $\mathcal{A}$'s Enc queries by sampling randomly, $\mathcal{E}_k$ queries using their own Enc oracle and forwarding any other queries.

It is important to observe that neither $\mathcal{B}$ nor $\mathcal{C}$ make prohibited or pointless queries. Any query $\mathcal{A}$ makes to his $\mathcal{E}_k$ oracle can be made by $\mathcal{B}$ to his Enc oracle, since if the $\mathcal{E}_k$ query was allowed, it must also be allowed for Enc. Conversely, any queries prohibited or pointless due to a previous Enc query are also prohibited or pointless if the equivalent query has been made to $\mathcal{E}_k$. (A prohibited query by $\mathcal{A}$ may become a pointless query for $\mathcal{B}$ or vice versa, but this detail doesn't worry us as effective adversaries make neither.) Thus any query that is prohibited or pointless for $\mathcal{B}$ was already prohibited or pointless for $\mathcal{A}$. The exact same argument shows $\mathcal{C}$ does not make prohibited or pointless queries.

Thus $\mathcal{B}$ simulates game (I) and $\mathcal{C}$ simulates (II): $\mathcal{B}$ wins if an only if $\mathcal{A}$ wins game (I), and $\mathcal{C}$ wins if and only if $\mathcal{A}$ wins game (II). Overall then,

$$\mathsf{Adv}^{1\mathtt{w}1\mathtt{yz}}_{\mathcal{E},\Lambda}(\mathcal{A}) \le \mathsf{Adv}^{\mathtt{v}1\mathtt{x}0\mathtt{z}}_{\mathcal{E},\Lambda}(\mathcal{B}) + \mathsf{Adv}^{\mathtt{v}1\mathtt{x}0\mathtt{z}}_{\mathcal{E},\Lambda}(\mathcal{C}) \le 2 \cdot \mathsf{Adv}^{1\mathtt{w}0\mathtt{yz}}_{\mathcal{E},\Lambda} ,$$

showing that, at the cost of a factor of 2, the additional $\mathcal{E}_k$ oracle does not help.

The proof of the decryption result is equivalent, using the intermediate world $(\mathcal{E}_k, \perp, \mathcal{E}_k, \perp, \Lambda_k)$. Let $\mathcal{A}$ be an adversary against the game $\mathtt{v}1\mathtt{x}1\mathtt{z}$, where only the Dec and $\mathcal{D}_k$ oracles are necessarily available to the adversary, with access to the others dependent on $\mathtt{vxz}$. By triangle inequality,

$$\mathsf{Adv}^{\mathtt{v}1\mathtt{x}1\mathtt{z}}_{\mathcal{E},\Lambda}(\mathcal{A}) = \Delta^{\mathcal{E}_k,\mathcal{D}_k,\mathcal{E}_k,\mathcal{D}_k,\Lambda_k}_{\$,\perp,\mathcal{E}_k,\mathcal{D}_k,\Lambda_k}(\mathcal{A}) \le \overbrace{\Delta^{\mathcal{E}_k,\mathcal{D}_k,\mathcal{E}_k,\mathcal{D}_k,\Lambda_k}_{\$,\perp,\mathcal{E}_k,\perp,\Lambda_k}(\mathcal{A})}^{\text{(I)}} + \overbrace{\Delta^{\$,\perp,\mathcal{E}_k,\perp,\Lambda_k}_{\$,\perp,\mathcal{E}_k,\mathcal{D}_k,\Lambda_k}(\mathcal{A})}^{\text{(II)}}.$$

Let $\mathcal{B}$ and $\mathcal{C}$ be adversaries against $\mathsf{Adv}^{\mathtt{v}1\mathtt{x}0\mathtt{z}}_{\mathcal{E},\Lambda}$ who reach their answer by running $\mathcal{A}$ and forwarding $\mathcal{A}$'s answer on as their own (and therefore running within similar resources to $\mathcal{A}$). This time, $\mathcal{B}$ answers $\mathcal{A}$'s $\mathcal{D}_k$ queries using his own Dec oracle and forwarding all other queries, whereas $\mathcal{C}$ answers $\mathcal{A}$'s Dec queries with $\perp$, $\mathcal{D}_k$ queries using their own Dec oracle and forwards any other queries. Since the same restrictions on prohibited or pointless queries apply to both Dec and $\mathcal{D}_k$, $\mathcal{B}$ and $\mathcal{C}$ both respect these requirements because $\mathcal{A}$ did.

Thus $\mathcal{B}$ simulates game (I) and $\mathcal{C}$ simulates (II), and so the advantage of $\mathcal{A}$ is at most twice that of the optimal adversary against game $1\mathtt{w}0\mathtt{yz}$. Thus, at the cost of a factor of 2, the additional $\mathcal{D}_k$ oracle does not help the adversary. □

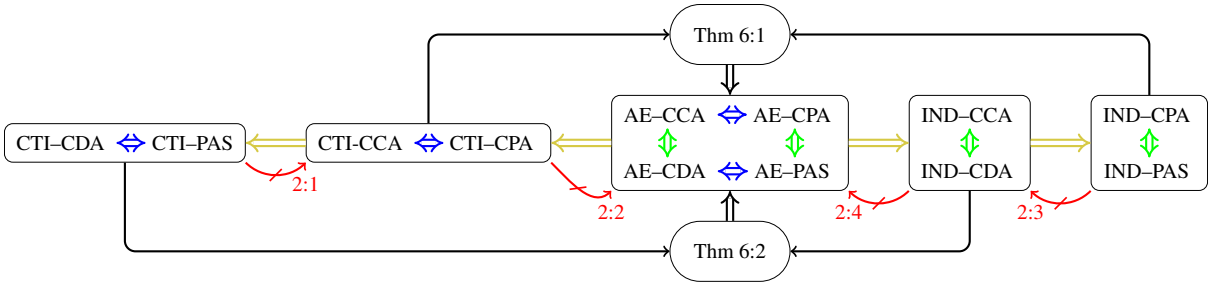

Fig. 3: The relations and equivalences between the 12 notions of leakage-free security. Each separation (red) is labelled with the appropriate sub-result of Lemma 2 that proves it. The horizontal implications (yellow) are given by Proposition 1:1, vertical equivalences (green) by 1:2, and horizontal equivalences (blue) by 1:3. The two methods for proving AE security are proven in Theorem 6. The separations between the subtle notions are equivalent, with the subtle and non-subtle worlds completely separated.

The implications given in Proposition 1 when restricted to classical scenario without leakage are visualised in Figure 3, along with two composition results from Theorem 6 and four basic separations (Lemma 2). Together, these results provide a complete characterization of the classical notions. An identical figure could be drawn for our subtle notions, where the subtle notions imply their non-subtle counterpart (Proposition 1) but not the other way around (Lemma 2).

Many of the separations are trivial or well known, with origins in folklore or the literature in the probabilistic LOR paradigm. In that setting, the equivalent of Lemma 2:3 is described as folklore by the original CPA paper [10] and that of Lemma 2:1 and 2:2 is assumed in the paper introducing INT–CTXT [14]. Finally, an equivalent to Lemma 2:4 was provided by Bellare and Namprempre [14, Section 3].

Separations are typically proven by assuming the existence of a scheme meeting the weaker notion and then adapting this scheme into one that still meets the weaker notion, yet is demonstrably insecure under the stronger notion. However, not every counterexample from the probabilistic LOR world carries over into the nonce-based formalisation with an ideal reference world that we are interested in. For example, we will later look at compound notions (Theorem 6) and see that the addition of passive integrity suffices to transform a nonce-based IND–CCA scheme into secure nonce-based AE scheme, despite the existence of an explicit counterexample in the probabilistic LOR–CCA setting [14]. For completeness, we therefore provide a comprehensive list of counterexamples defined within our framework.

For practical relevance, separations should employ counterexamples that are both tidy and length-regular. Our separations meet this criterium, unlike for instance the counterexample to Lemma 2:4 as provided by Bellare and Namprempre [14, Section 3] (0 has multiple encryptions).

From a theoretical perspective, separations ideally use the weakest possible assumption with counterexamples leaving all input spaces and the stretch invariant [50]. Our separations will be sloppy in the sense that we use the existence of a secure AE scheme as underlying assumption (which is stronger than necessary) and will not always leave input spaces or stretch invariant.

Lemma 2. *There are no generic implications between notions beyond those given in Propositon 1. In particular,*

1. *CTI–PAS* $\not\Rightarrow$ *CTI–CPA;*
2. *CTI–CPA* $\not\Rightarrow$ *AE–PAS;*
3. *IND–PAS* $\not\Rightarrow$ *IND–CCA;*
4. *IND–CCA* $\not\Rightarrow$ *AE–PAS.*

Proof. Since leakage could be trivial, separations in the leak-free case immediately extend to the subtle setting. Conversely, since the leakage could be catastrophic (e.g. $\Lambda_k = k$) even for an otherwise secure scheme, games with a leakage oracle cannot be generically equivalent to their non-leaky counterpart. Thus, given known implications, providing the four explicitly stated separations suffices.

Let $\mathcal{E}$ be a tidy, length-regular, secure AE scheme, then for each separation, we will build a related tidy and length-regular scheme that is secure under the weaker notion, yet insecure under the stronger notion. Assume that $\mathsf{K} \subseteq \mathsf{M}$, let $\mathbf{0} = 0^{|k|}$ be a string of zeros the same length as k, $\mathbf{1} = 1^{\tau(k)}$ a string of ones the same length as $\mathcal{E}_k^{\mathbf{0},\mathbf{0}}(k)$ and assume that $\mathbf{0} \in \mathsf{N}, \mathsf{A}$ and $\mathbf{1} \in \mathsf{C}$. (If this is not the case alternative strings can be used, there is nothing special about $\mathbf{0}$ or $\mathbf{1}$.)

(1) CTI–PAS $\not\Longrightarrow$ CTI–CPA. Consider the scheme $\tilde{\mathcal{E}}_k^{N,A}(M) := M||k$. A CTI–CPA adversary has access to an encryption oracle from which they can immediately recover the secret key and then trivially distinguish the challenge oracle. However, a CTI–PAS adversary only has their challenge oracle, which only returns $\perp$ until queried with a valid (N, A, C) triple. So, until they win the game, the adversary learns nothing from his oracle, and thus can only create a valid (N, A, C) by guessing the key, with probability $1/|\mathsf{K}|$ (per query). Thus the scheme is CTI–PAS secure but not CTI–CPA.

(2) CTI–CPA $\not\Longrightarrow$ AE–PAS. Define $\tilde{\mathcal{E}}_k^{N,A}(M) := 1||\mathcal{E}_k^{N,A}(M)$. In the AE–PAS game, an adversary can make a single Enc query and returne the first bit of the oracles response as his answer, to distinguish with probability $1/2$. In the CTI–CPA game, no Enc oracle is provided and any attack against $\tilde{\mathcal{E}}_k$ implies one against $\mathcal{E}_k$: $\tilde{\mathcal{E}}_k$ can be perfectly simulated with access to $\mathcal{E}_k$, and similarly $\widetilde{\mathrm{Dec}}$ with access to Dec. Thus CTI–CPA security of $\mathcal{E}$ implies the same level of CTI–CPA security of $\tilde{\mathcal{E}}$, meaning that $\tilde{\mathcal{E}}$ separates CTI–CPA and AE–PAS.

(3) IND–PAS $\not\Longrightarrow$ IND–CCA. We will take a secure scheme $\mathcal{E}$, and modify it to form $\tilde{\mathcal{E}}$ by using a second key l to construct (for each key k) two special points that can be discovered and exploited by an adversary with access to $\tilde{\mathcal{D}}_{k||l}$ (e.g. under IND–CCA). They will use the first special point to learn an unusual property of $\tilde{\mathcal{E}}$, and the second to validate if this was correct or not. However, these points will be defined in a way that an adversary with access only to $\tilde{\mathcal{E}}_{k||l}$ cannot detect them, preserving IND–PAS security.

Let $l \leftarrow\!\!\$\ \mathsf{K}$, and define $\tilde{\mathcal{E}}_k$ by

$$\tilde{\mathcal{E}}_{k||l}^{N,A}(M) := \begin{cases} 0^{|\mathbf{1}|} & (N,A,M) = (\mathbf{0},\mathbf{0},l) \\ \mathbf{1} & (N,A,M) = (\mathbf{0},\mathbf{0},l \oplus 1) \\ \gamma_k & (N,A,M) = (\mathbf{0},\mathbf{0},\alpha_k), \alpha_k \neq \perp \\ \delta_k & (N,A,M) = (\mathbf{0},\mathbf{0},\beta_k), \beta_k \neq \perp \\ \mathcal{E}_k^{N,A}(M) & \text{Otherwise} \end{cases},$$

where $\alpha_k, \beta_k, \gamma_k, \delta_k$ are chosen for each key to ensure the function is injective, and thus correct. Explicitly, let $\alpha_k = \mathcal{D}_k^{\mathbf{0},\mathbf{0}}(0^{|\mathbf{1}|})$, $\beta_k = \mathcal{D}_k^{\mathbf{0},\mathbf{0}}(\mathbf{1})$, $\gamma_k = \mathcal{E}_k^{\mathbf{0},\mathbf{0}}(l)$ and $\delta_k = \mathcal{E}_k^{\mathbf{0},\mathbf{0}}(l \oplus 1)$. Then, if $\alpha_k = l \oplus 1$ or $\beta_k = 1$, swap the values of γ_k and δ_k. This makes $\tilde{\mathcal{E}}$ a well defined function between the appropriate spaces, tidy and correct, and thus an AE scheme.

Consider first the IND–CCA case, and adversary $\mathcal{A}$ who acts as follows. First, $\mathcal{A}$ makes a single decryption query to learn $l \leftarrow \tilde{\mathcal{D}}_{k||l}^{\mathbf{0},\mathbf{0}}(0^{|\mathbf{1}|})$. Having done so, they then query $\widetilde{\mathrm{Enc}}^{\mathbf{0},\mathbf{0}}(l \oplus 1)$ and return 1 if this output is $\mathbf{1}$. In the real case, $\mathcal{A}$ will always output 1, and in the ideal case they will output 1 with probability $1/2^{\tau(k)}$, which is small (else an adversary against the IND–PAS security of $\mathcal{E}$ could simply guess the key). Thus $\mathcal{A}$ distinguishes $\tilde{\mathcal{E}}$ under the IND–CCA game.

However, in the IND–PAS setting (where an adversary has access to just an $\widetilde{\mathrm{Enc}}$ oracle) the scheme inherits security from $\mathcal{E}$. Since $\tilde{\mathcal{E}}$ only differs from $\mathcal{E}$ on a few specific inputs, the adversarial advantage in distinguishing between $\mathcal{E}$ and $\tilde{\mathcal{E}}$ is at most the probability of querying one of these 4 inputs. So, for each of these special points we now bound this probability, assuming the adversary has not previously queried any of the others to upper bound the overall advantage.

As l is sampled uniformly and independently of k, the probability of an adversary querying $\tilde{\mathcal{E}}_{k||l}{}^{\mathbf{0},\mathbf{0}}(l)$ or $\tilde{\mathcal{E}}_{k||l}^{\mathbf{0},\mathbf{0}}(l \oplus 1)$ is at most $q/2^{|k|-1}$ for an adversary making q queries. To bound the probability of an adversary querying α_k or β_k, we will instead bound the probability that these special points exist at all. Consider the adversary $\mathcal{B}$ against the AE security of $\mathcal{E}$ that makes two queries: $\widetilde{\mathrm{Dec}}^{\mathbf{0},\mathbf{0}}(0^{|\mathbf{1}|})$ and $\widetilde{\mathrm{Dec}}^{\mathbf{0},\mathbf{0}}(\mathbf{1})$. If both are $\perp$, $\mathcal{B}$ returns 0, and otherwise $\mathcal{B}$ returns 1. Thus $\mathcal{B}$ returns 1 if either α_k or β_k is

not $\perp$, and so the probability of these special points existing is bounded by that of $\mathcal{B}$ winning the AE game against $\mathcal{E}$.

So, together these events bound the difference between $\mathcal{E}_k$ and $\tilde{\mathcal{E}}_{k||l}$, meaning that

$$\begin{aligned}\mathbb{P}\left[\mathcal{A}^{\tilde{\mathcal{E}}_{k||l}} \rightarrow 1\right] - \mathbb{P}\left[\mathcal{A}^{\mathcal{E}_k} \rightarrow 1\right] & \\ \leq \mathbb{P}\left[\mathcal{A}^{\tilde{\mathcal{E}}_{k||l}} \text{ queries } (\mathbf{0},\mathbf{0},l) \text{ or } (\mathbf{0},\mathbf{0},l\oplus 1)\right] &+ \mathbb{P}\left[\mathcal{A}^{\tilde{\mathcal{E}}_{k||l}} \text{ queries } (\mathbf{0},\mathbf{0},\alpha) \text{ or } (\mathbf{0},\mathbf{0},\beta)\right] \\ \leq \qquad q/2^{|k|-1} &+ \mathsf{Adv}^{\mathrm{AE}}_{\mathcal{E}}(\mathcal{B}) \qquad .\end{aligned}$$

So,

$$\mathsf{Adv}^{\text{IND–PAS}}_{\tilde{\mathcal{E}}}(\mathcal{A}) \leq \mathsf{Adv}^{\text{IND–PAS}}_{\mathcal{E}}(\mathcal{A}) + q/2^{|k|-1} + \mathsf{Adv}^{\mathrm{AE}}_{\mathcal{E}}(\mathcal{B}).$$

Since $\mathcal{E}$ was (by assumption) AE secure, these terms are all small, implying the IND–PAS security of $\tilde{\mathcal{E}}$.

(4) IND–CCA $\not\Longrightarrow$ AE–PAS. We will define $\tilde{\mathcal{E}}$ to be an IND–CCA scheme with double the keyspace of $\mathcal{E}$ and interpret this double-length key as $k||l$. Then define $\alpha_k := \mathcal{D}_k^{\mathbf{0},\mathbf{0}}(\mathbf{1})$ and $\beta_k := \mathcal{E}_k^{\mathbf{0},\mathbf{0}}(l)$.

$$\tilde{\mathcal{E}}^{N,A}_{k||l}(M) := \begin{cases} \mathbf{1} & (N,A,M) = (\mathbf{0},\mathbf{0},l) \\ \beta_k & (N,A,M) = (\mathbf{0},\mathbf{0},\alpha_k) \wedge \alpha_k \neq \perp \\ \mathcal{E}_k^{N,A}(M) & \text{Otherwise} \end{cases}$$

So, for each key, $\tilde{\mathcal{E}}$ is the same as $\mathcal{E}$ except with the preimages of $\mathbf{1}$ and β exchanged under nonce-associated data pair $(\mathbf{0},\mathbf{0})$. Clearly $(\mathbf{0},\mathbf{0},\mathbf{1})$ is a valid ciphertext for any key, and thus in the AE–PAS game the scheme is trivially distinguishable by testing $(\mathbf{0},\mathbf{0},\mathbf{1})$ at the Dec oracle. However, as we will now show, the scheme remains secure in the IND–CCA game.

Let $\mathcal{A}$ be an adversary against $\tilde{\mathcal{E}}$ in the IND–CCA game making at most q queries. Using the same logic as for the previous case, there exists an adversary $\mathcal{B}$ against $\mathcal{E}$ who wins the AE game if $\alpha_k \neq \perp$, and so we restrict ourselves to the case $\alpha_k = \perp$. With this restriction, $\tilde{\mathcal{E}}$ acts identically to $\mathcal{E}$ unless $\mathcal{A}$ queries $\mathrm{Enc}(\mathbf{0},\mathbf{0},l)$ or $\mathrm{Dec}(\mathbf{0},\mathbf{0},\mathbf{1})$. Note that he cannot make both of these queries since, after making the first, the second is prohibited.

Making a query $\mathrm{Enc}(\mathbf{0},\mathbf{0},l)$ corresponds to guessing l, and so occurs with probability at most $q/|\mathsf{K}|$.

An adversary who queries $\mathrm{Dec}(\mathbf{0},\mathbf{0},\mathbf{1})$ learns l. However, knowledge of l does not assist the adversary in distinguishing the scheme, since l is sampled uniformly at random, independent from k, and does not affect any other elements of $\tilde{\mathcal{E}}_k$ beyond those already queried or now prohibited. So, as long as $\alpha_k = \perp$ and the adversary does not query $\mathrm{Enc}(\mathbf{0},\mathbf{0},l)$, the probability of the adversary winning is simply that of $\mathcal{A}$ winning the AE game against $\mathcal{E}$.

Over all,

$$\begin{aligned}\mathsf{Adv}^{\text{IND–CCA}}_{\tilde{\mathcal{E}}_k,\tilde{\mathcal{D}}_k}(\mathcal{A}) \leq \;& \mathbb{P}\left[\alpha_k \neq \perp\right] + \mathbb{P}\left[\mathcal{A}^{\tilde{\mathcal{E}}} \text{ queries } \mathrm{Enc}(\mathbf{0},\mathbf{0},l)\right] + \mathsf{Adv}^{\text{IND–CCA}}_{\mathcal{E}_k,\mathcal{D}_k}(\mathcal{A}) \\ \leq \;& \mathsf{Adv}^{\mathrm{AE}}_{\mathcal{E}_k,\mathcal{D}_k}(\mathcal{B}) + \qquad q/|\mathsf{K}| \qquad + \mathsf{Adv}^{\text{IND–CCA}}_{\mathcal{E}_k,\mathcal{D}_k}(\mathcal{A})\end{aligned}$$

each of which is small because $\mathcal{E}$ was a secure AE scheme.

□

Recovering classical notions. Reassuringly, the classical setting can be recovered by setting the fifth bit (leakage) to be zero, thereby simply ignoring the leakage oracle.

Our IND–CPA and IND–PAS notions are equivalent; the latter is the common definition for chosen-plaintext confidentiality in the modern literature, where it is often referred to as IND–CPA or even just CPA (e.g. [4]). Similarly, our IND–CCA definition and IND–CDA are equivalent, with the latter matching the original nonce-based IND$–CCA definition [48]. That said, the relationship between IND$–CCA and LOR–CCA is rather complex, especially for tidy schemes, a topic discussed further in Appendix B.

To formalize integrity of ciphertexts, both the CTI–CCA notion and the equivalent CTI–CPA have appeared in the past (e.g. [31] or [1] respectively); it was already known that for ciphertext integrity the two notions coincide, though not for plaintext integrity [11].

The strongest authenticated encryption notion is AE–CCA which, as there is no benefit for the adversary in having the additional honest oracles, is equivalent to AE–PAS. Following Rogaway and Shrimpton [51], the modern definition of Authenticated Encryption has been precisely AE–PAS, and so we follow the literature in referring to all the AE goals (but particularly AE–PAS) simply as AE.

Concluding, our framework (re)confirms the use of existing definitions, providing a seemless extension to incorporate (deterministic) decryption leakage.

Simplified games and adversaries. Proposition 1 and Lemma 2 demonstrate that a number (but not all) of the games are equivalent. For each equivalence class there is a unique game with the least amount of oracle access to an adversary; we refer to this as the simplified game. Thus we can restate the previous results in the following corollary:

Corollary 3. *The simplified games are just 1100z, 1000z, 1001z, 0110z, and 0100z (where z signifies a bit that might or might not be set). These correspond to* AE–PAS, IND–PAS, IND–CDA, CTI–CPA, *and* CTI–PAS*, along with their subtle variants.*

Henceforth, we will consider only simplified games under effective adversaries (i.e. those that do not make any pointless queries). In the simplified games an adversary is effectively presented with (at most) three oracles, with access to (at most) one oracle following the syntax of encryption, decryption or leakage. We use the notation $\Delta^{\mathcal{O}_1,\mathcal{O}_2,\mathcal{O}_3}_{\mathcal{O}_a,\mathcal{O}_b,\mathcal{O}_c}$ as short-hand for the adversarial advantage in distinguishing between oracles $1, 2, 3$ and oracles a, b, c. Here the first oracle will always correspond to encryption, the second to decryption, and the third to leakage. We use $\emptyset$ to denote that an oracle is not available. Although what constitute pointless and prohibited queries does not follow from the notation, the concepts are still captured as we only consider effective adversaries (obviously, in reductions we will check 'effectiveness' preservation as appropriate).

2.5 Security of Subtle Authenticated Encryption

Recall that we defined an SAE scheme as a pair $(\mathcal{E}, \Lambda)$ (along with appropriate spaces), or equivalently the triple $(\mathcal{E}, \mathcal{D}, \Lambda)$. When we talk of a *secure* SAE scheme, we mean that the AE–sCCA advantage is sufficiently small. In other words, even if the adversary has access to honest encryption and decryption oracles as well as clearly specified leakage from the decryption function, she can neither distinguish real from random ciphertexts (distinct from those output by the honest encryption oracle) nor forge ciphertexts. This characterisation closely matches the real-world perspective, where an adversary might have adaptive access to both encryption and leaky decryption functionalities, yet it gains no advantage over any ciphertexts not trivially compromised.

In Section 3 we will provide a comparison of secure SAE and the earlier BDPS, RUP, and RAE definitions. In particular, the inspiration for the remainder of this section will become a lot clearer—we are especially indebted to the BDPS and RUP papers.

While AE–sCCA (or 11111) is the strongest goal describable within our framework, due to Proposition 1, it suffices from the designer's point of view to demonstrate that the scheme is AE–sPAS (11001), restricted to adversaries who make neither pointless nor prohibited queries. There are many ways one may approach proving SAE security, and one of the most intuitive is to reduce the problem to the leakage-free case. That is, (i) prove that the scheme is a secure AE scheme in the absence of leakage and then (ii) show leakage is 'computationally independent' of the key against adversaries with access to honest encryption and decryption oracles. Before providing such a decomposition result (Theorem 6), we formalise what we mean by (ii) above.

Error simulatability. So far, all the games begin by picking a key $k \leftarrow\!\!\$\ \mathsf{K}$, and provide the adversary with access to some subset of an honest encryption oracle $\mathcal{E}_k$, honest decryption oracle $\mathcal{D}_k$, or honest leakage oracle Λ_k. All these oracles use the same key k. To argue that the leakage is harmless, we compare this set of oracles with one where the leakage is based on an independently, randomly sampled key $l \leftarrow\!\!\$\ \mathsf{K}$ instead. Thus, we define the goal ERR (for *Error Simulatability*) as that of distinguishing Λ_k from Λ_l, where k and l are sampled independently. The name and abbreviation ERR was inspired by the error invariance notion (INV–ERR) introduced by BPDS (see Section 3.1), while the game borrows from RUP's decryption independence (DI) (see Section 3.2).

As always, this goal can be paired with any set of powers (PAS, CPA, CDA, CCA), where the encryption and decryption oracles always use k (never l). The strongest such security notion then is ERR–CCA:

$$\mathsf{Adv}^{\text{ERR–CCA}}_{\mathcal{E},\Lambda} := \Delta^{\mathcal{E}_k,\mathcal{D}_k,\Lambda_k}_{\mathcal{E}_k,\mathcal{D}_k,\Lambda_l}$$

The adversary cannot forward an output triple from the encryption oracle to the leakage oracle, since the real leakage oracle would return $\top$, whereas the simulated version would almost certainly not. That is, if they have previously queried $\mathcal{E}_k(N, A, M) = C$, they may not query $\Lambda(N, A, C)$. Any other sequence of queries is allowed, although clearly repeating queries between $\mathcal{E}_k$ and $\mathcal{D}_k$ would be pointless. It is important to note that the adversary against SAE meets this requirement, thus upper bounds on the error invariance of a scheme can be used towards proving overall security.

Simulator-free definitions. At first sight, our choice of simulator may appear unnecessarily specific: why not use a definition with regards to an arbitrary simulator and then leave it to the designer of the scheme to come up with a good one? Well, as Lemma 4 shows, if there exists any good simulator, then Λ_l is one as well. A similar observation was made in RUP, where PA2 is shown equivalent to DI [3, Thms. 8,9]. The benefit of fixing the simulator to Λ_l is that security is completely described by $(\mathcal{E}, \mathcal{D}, \Lambda)$—or even $(\mathcal{E}_k, \Lambda_k)$ as $\mathcal{D}_k$ is wholly defined by $\mathcal{E}_k$—rather than requiring an additional quantification over or dependence on some simulator. Obviously to simplify a proof, one could still use an alternative simulator.

Lemma 4. *Let* S *be an arbitrary stateful simulator (without oracle access). Then*

$$\mathsf{Adv}^{\text{ERR–CCA}}_{\mathcal{E},\Lambda} \leq 2 \cdot \Delta^{\mathcal{E}_k,\mathcal{D}_k,\Lambda_k}_{\mathcal{E}_k,\mathcal{D}_k,\,\mathsf{S}}$$

moreover there exists a simulator S *with* $\Delta^{\mathcal{E}_k,\mathcal{D}_k,\Lambda_k}_{\mathcal{E}_k,\mathcal{D}_k,\,\mathsf{S}} = \mathsf{Adv}^{\text{ERR–CCA}}_{\mathcal{E},\Lambda}$.

Proof. Since the simulator S has no oracle access, it is necessarily independent of the game's key k. So, the adversarial advantage in distinguishing between S and Λ_k is precisely the same as that from S to Λ_l: the difference is merely a relabelling. That is, $\Delta^{\mathcal{E}_k,\mathcal{D}_k,\,\mathsf{S}}_{\mathcal{E}_k,\mathcal{D}_k,\Lambda_l} = \Delta^{\emptyset,\emptyset,\Lambda_l}_{\emptyset,\emptyset,\,\mathsf{S}} = \Delta^{\emptyset,\emptyset,\Lambda_k}_{\emptyset,\emptyset,\,\mathsf{S}}$, and thus

$$\begin{aligned}\mathsf{Adv}^{\text{ERR–CCA}}_{\mathcal{E},\Lambda} := \Delta^{\mathcal{E}_k,\mathcal{D}_k,\Lambda_k}_{\mathcal{E}_k,\mathcal{D}_k,\Lambda_l} &\leq \Delta^{\mathcal{E}_k,\mathcal{D}_k,\Lambda_k}_{\mathcal{E}_k,\mathcal{D}_k,\,\mathsf{S}} + \Delta^{\mathcal{E}_k,\mathcal{D}_k,\,\mathsf{S}}_{\mathcal{E}_k,\mathcal{D}_k,\Lambda_l} \\ &= \Delta^{\mathcal{E}_k,\mathcal{D}_k,\Lambda_k}_{\mathcal{E}_k,\mathcal{D}_k,\,\mathsf{S}} + \Delta^{\emptyset,\emptyset,\Lambda_k}_{\emptyset,\emptyset,\,\mathsf{S}} \quad \leq 2 \cdot \Delta^{\mathcal{E}_k,\mathcal{D}_k,\Lambda_k}_{\mathcal{E}_k,\mathcal{D}_k,\,\mathsf{S}}\ .\end{aligned}$$

The inverse result is trivial, since we may choose $\mathsf{S} := \Lambda_l$. □

Initial observations on error simulatability. Let's consider briefly how hard these new goals are to achieve. We immediately have that $\mathsf{Adv}^{\text{ERR–PAS}}_{\mathcal{E},\Lambda} = 0$, as the adversary is bereft of any guaranteed access to the real key k to compare the challenge oracle Λ_k vs. Λ_l against. In other words, any scheme is perfectly ERR–PAS secure.

Since we do not prohibit the adversary from making the same query to both their leakage and decryption oracles, there exists a forgery-based attack against the ERR–CCA (or ERR–CDA) game. The adversary creates a forgery, which they then send to both their decryption and leakage oracles. If the forgery if valid, $\mathcal{D}_k$ will output a message rather than $\perp$. Then, if the leakage oracle is Λ_k it will output

$\top$, whereas Λ_l will almost certainly output some $\perp_i \in \mathsf{L}$, distinguishing the real from the simulated leakage.

Indeed, one may wonder whether the notions do not overlap, whether ERR–CCA implies some CTI notion, but this is not the case as summarized in Lemma 5. Using the implications of Proposition 1, it follows that the error invariance notions neither imply nor are implied by the non-subtle integrity or confidentiality notions. Nonetheless, in Section 2.6 we will see that ERR–CCA is likely to be achieved alongside some integrity notion.

Lemma 5 (Separating ERR notions from AE). *ERR–CCA implies neither CTI–PAS nor IND–PAS, and AE implies neither ERR–CPA nor ERR–CDA.*

Proof (sketch). (ERR–CCA $\not\Longrightarrow$ CTI–PAS) Consider a scheme for which every ciphertext is valid. This scheme is ERR–CCA secure and yet CTI–PAS insecure: winning the CTI–PAS game is trivial, yet the ERR–CCA advantage is zero (since the two worlds that need distinguishing are identical).

(ERR–CCA $\not\Longrightarrow$ IND–PAS) Consider the scheme where $\mathcal{E}_k(M) := 0||M$ and $\Lambda_k = 0$. Then $\Lambda_k = \Lambda_l$, yet ciphertexts are clearly distinguishable from random. Thus the scheme is ERR–CCA secure but not IND–PAS.

(AE $\not\Longrightarrow$ ERR–CPA/CDA) Consider a secure AE scheme, and suppose that $\Lambda_k := k$. The subtle scheme will still be AE-secure, but trivially insecure under the ERR–CPA or ERR–CDA games. □

2.6 Decomposing SAE security

The separations of Lemmas 2 and 5 suggest a sort of 'orthogonality' between the goals of indistinguishability, ciphertext integrity, and error invariance. In Figure 4 we visualise this as a cube, where each vertex corresponds to a world with which an adversary may be interacting, with the front-bottom-left corresponding with the real world, and the back-top-right corresponding to a fully idealised world. An edge between two vertices corresponds to the distinguishing advantage between the two corresponding worlds.

The three goals IND, CTI, and ERR each correspond to an axis in the diagram, such that the parallel edges represent the same goal, but under a different set of powers. Moving away from the front-bottom-left corresponds to removing adversarial access to one or more oracles. So any parallel edges in this direction represent the same notion except with fewer powers, and thus security of the second is implied (Proposition 1:1). The front face contains the new subtle notions, whereas the aft face corresponds to the traditional, leak-free view of authenticated encryption.

Diagonals in the cube represent compound notions such as AE and SAE. Since security notions are defined in terms of the difference between two possible worlds, these can instead be written as a sequence of security notions representing any alternative path over the cube from begin to end vertex (of the compound notion). This approach immediately visualizes possible composition results; their validity can easily be verified through use of the triangle inequality (and a check that effectiveness of adversaries is indeed preserved).

So, Figure 4 visualises the security notions and relations between them. Security of a notion on an edge implies security for any notions on parallel edges that are right, above or behind it. Any path yields a decomposition result for the compound notion defined by its endpoints, which we can prove with the triangle inequality. Moreover, since providing access to additional real oracles can only assist the adversary, for any path in which each node is closer to the ideal world than its predecessor, the converse also holds. Pulling these observations together, we form Theorem 6. The proofs for the enumerated examples are intuitive, and provided in their own short proofs.

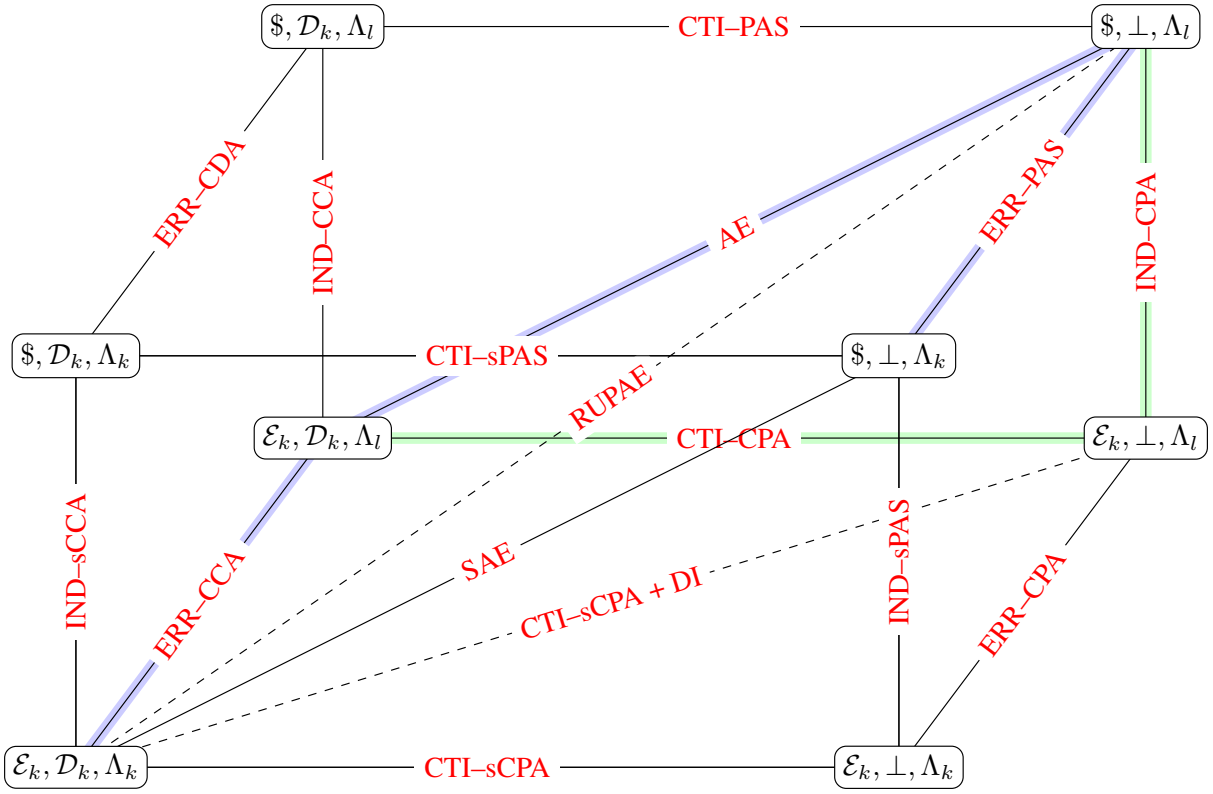

Fig. 4: A "cube" of notions, visualising SAE security as distances in 3-dimensional space. Each vertex denotes a world with which the adversary may be interacting, with edges representing security notions. The 12 edges of the cube are the CTI, IND and ERR notions, with diagonal edges denoting compound notions such as SAE: alternative paths between two nodes indicate a possible composition result. The highlighted paths represent useful decompositions of SAE (blue) and AE (green). The dotted lines denote RUPAE and the combination of CTI-sCPA+DI, notions associated with RUP (see Section 3.2).

Theorem 6 (Cube Theorem). *Consider the cube in Figure 4. Any path yields a decomposition result for the notion defined by its endpoints. For any path in which each node is closer to the ideal world than its predecessor the converse also holds, and security of the compound notion implies security of the two constituent notions. In particular,*

1. *AE* $\iff$ *IND–CPA + CTI–CPA*
2. *AE* $\iff$ *IND–CCA + CTI–PAS*
3. *SAE* $\iff$ *ERR–CCA + AE* $\iff$ *ERR–CCA + CTI–CPA + IND–CPA*
4. *ERR–CCA + CTI–CPA* $\iff$ *CTI–sCPA + ERR–CPA.*

Proof (Of Theorem 6:1 and 6:2). By Proposition 1, AE security implies CTI-PAS and IND-CPA security. Then, applying the triangle inequality:

$$\begin{aligned}
\mathsf{Adv}^{\text{IND–CCA}}_{\mathcal{E},\Lambda} &= \Delta^{\mathcal{E}_k,\mathcal{D}_k,\emptyset}_{\$,\mathcal{D}_k,\emptyset} \le \Delta^{\mathcal{E}_k,\mathcal{D}_k,\emptyset}_{\$,\perp,\emptyset} + \Delta^{\$,\perp,\emptyset}_{\$,\mathcal{D}_k,\emptyset} \le \mathsf{Adv}^{\text{AE}}_{\mathcal{E},\Lambda} + \mathsf{Adv}^{\text{CTI–PAS}}_{\mathcal{E},\Lambda} \le 2 \cdot \mathsf{Adv}^{\text{AE}}_{\mathcal{E},\Lambda} \\
\mathsf{Adv}^{\text{CTI–CPA}}_{\mathcal{E},\Lambda} &= \Delta^{\mathcal{E}_k,\mathcal{D}_k,\emptyset}_{\mathcal{E}_k,\perp,\emptyset} \le \Delta^{\mathcal{E}_k,\mathcal{D}_k,\emptyset}_{\$,\perp,\emptyset} + \Delta^{\$,\perp,\emptyset}_{\mathcal{E}_k,\perp,\emptyset} \le \mathsf{Adv}^{\text{AE}}_{\mathcal{E},\Lambda} + \mathsf{Adv}^{\text{IND–CPA}}_{\mathcal{E},\Lambda} \le 2 \cdot \mathsf{Adv}^{\text{AE}}_{\mathcal{E},\Lambda} \\
\mathsf{Adv}^{\text{AE}}_{\mathcal{E},\Lambda} &= \Delta^{\mathcal{E}_k,\mathcal{D}_k,\emptyset}_{\$,\mathcal{D}_k,\emptyset} \le \Delta^{\mathcal{E}_k,\mathcal{D}_k,\emptyset}_{\mathcal{E}_k,\mathcal{D}_k,\emptyset} + \Delta^{\mathcal{E}_k,\mathcal{D}_k,\emptyset}_{\$,\perp,\emptyset} \le \mathsf{Adv}^{\text{IND–CCA}}_{\mathcal{E},\Lambda} + \mathsf{Adv}^{\text{CTI–PAS}}_{\mathcal{E},\Lambda} \\
\mathsf{Adv}^{\text{AE}}_{\mathcal{E},\Lambda} &= \Delta^{\mathcal{E}_k,\mathcal{D}_k,\emptyset}_{\$,\perp,\emptyset} \le \Delta^{\mathcal{E}_k,\mathcal{D}_k,\emptyset}_{\mathcal{E}_k,\perp,\emptyset} + \Delta^{\mathcal{E}_k,\perp,\emptyset}_{\$,\perp,\emptyset} \le \mathsf{Adv}^{\text{CTI–CPA}}_{\mathcal{E},\Lambda} + \mathsf{Adv}^{\text{IND–CPA}}_{\mathcal{E},\Lambda}
\end{aligned}$$

□

Proof (Of Theorem 6:3). Having first made the immediate simplifications of Proposition 1, we proceed by triangle inequality:

$$\begin{aligned}
\mathsf{Adv}^{\text{SAE}}_{\mathcal{E},\Lambda} = \Delta^{\mathcal{E}_k,\mathcal{D}_k,\Lambda_k}_{\$,\perp,\Lambda_k} &\le \Delta^{\mathcal{E}_k,\mathcal{D}_k,\Lambda_k}_{\mathcal{E}_k,\mathcal{D}_k,\Lambda_l} + \Delta^{\mathcal{E}_k,\mathcal{D}_k,\Lambda_l}_{\$,\perp,\Lambda_l} + \Delta^{\$,\perp,\Lambda_l}_{\$,\perp,\Lambda_k} \\
&= \mathsf{Adv}^{\text{ERR–CCA}}_{\mathcal{E},\Lambda} + \mathsf{Adv}^{\text{AE}}_{\mathcal{E},\Lambda} + \mathsf{Adv}^{\text{ERR–PAS}}_{\mathcal{E},\Lambda}
\end{aligned}$$

Since the ERR–PASS game is perfectly secure, security in the AE and ERR–CCA games implies SAE security. Conversely,

$$\begin{aligned}
\mathsf{Adv}^{\mathrm{AE}}_{\mathcal{E},\Lambda} &= \Delta^{\mathcal{E}_k,\mathcal{D}_k,\emptyset}_{\$\,,\,\perp\,,\emptyset} \le \Delta^{\mathcal{E}_k,\mathcal{D}_k,\Lambda_k}_{\$\,,\,\perp\,,\Lambda_k} && && \le \mathsf{Adv}^{\mathrm{SAE}}_{\mathcal{E},\Lambda}\\
\mathsf{Adv}^{\mathrm{ERR\text{–}CCA}}_{\mathcal{E},\Lambda} &= \Delta^{\mathcal{E}_k,\mathcal{D}_k,\Lambda_k}_{\mathcal{E}_k,\mathcal{D}_k,\Lambda_l} \le \Delta^{\mathcal{E}_k,\mathcal{D}_k,\Lambda_k}_{\$\,,\,\perp\,,\Lambda_k} &+ \Delta^{\$,\perp,\Lambda_k}_{\$,\perp,\Lambda_l} &+ \Delta^{\$\,,\,\perp\,,\Lambda_l}_{\mathcal{E}_k,\mathcal{D}_k,\Lambda_l}\\
&= \mathsf{Adv}^{\mathrm{SAE}}_{\mathcal{E},\Lambda} &+ \mathsf{Adv}^{\mathrm{ERR\text{–}PAS}}_{\mathcal{E},\Lambda} &+ \mathsf{Adv}^{\mathrm{AE}}_{\mathcal{E},\Lambda} \le 2\cdot\mathsf{Adv}^{\mathrm{SAE}}_{\mathcal{E},\Lambda}
\end{aligned}$$

where again we have used that $\mathsf{Adv}^{\mathrm{ERR\text{–}PAS}}_{\mathcal{E},\Lambda} = 0$. □

Proof (Of Theorem 6:4). Immediately, we have that CTI–CPA is implied by CTI–sCPA, and ERR–CPA by ERR–CCA. The other two implications can be shown by triangle inequality:

$$\begin{aligned}
\mathsf{Adv}^{\mathrm{CTI\text{–}sCPA}}_{\mathcal{E},\Lambda} = \Delta^{\mathcal{D}_k,\mathcal{E}_k,\Lambda_k}_{\perp\,,\mathcal{E}_k,\Lambda_k} &\le \Delta^{\mathcal{D}_k,\mathcal{E}_k,\Lambda_k}_{\mathcal{D}_k,\mathcal{E}_k,\Lambda_l} &+ \Delta^{\mathcal{D}_k,\mathcal{E}_k,\Lambda_l}_{\perp\,,\mathcal{E}_k,\Lambda_l} &+ \Delta^{\perp,\mathcal{E}_k,\Lambda_l}_{\perp,\mathcal{E}_k,\Lambda_k}\\
&\le \Delta^{\mathcal{E}_k,\mathcal{D}_k,\Lambda_k}_{\mathcal{E}_k,\mathcal{D}_k,\Lambda_l} &+ \Delta^{\mathcal{D}_k,\mathcal{E}_k,\emptyset}_{\perp\,,\mathcal{E}_k,\emptyset} &+ \Delta^{\mathcal{D}_k,\mathcal{E}_k,\Lambda_l}_{\mathcal{D}_k,\mathcal{E}_k,\Lambda_k}\\
&= 2\cdot\mathsf{Adv}^{\mathrm{ERR\text{–}CCA}}_{\mathcal{E},\Lambda} &+ \mathsf{Adv}^{\mathrm{CTI\text{–}CPA}}_{\mathcal{E},\Lambda}\\
\mathsf{Adv}^{\mathrm{ERR\text{–}CCA}}_{\mathcal{E},\Lambda} = \Delta^{\mathcal{E}_k,\mathcal{D}_k,\Lambda_k}_{\mathcal{E}_k,\mathcal{D}_k,\Lambda_l} &\le \Delta^{\mathcal{E}_k,\mathcal{D}_k,\Lambda_k}_{\mathcal{E}_k,\,\perp\,,\Lambda_k} &+ \Delta^{\mathcal{E}_k,\perp,\Lambda_k}_{\mathcal{E}_k,\perp,\Lambda_l} &+ \Delta^{\mathcal{E}_k,\,\perp\,,\Lambda_l}_{\mathcal{E}_k,\mathcal{D}_k,\Lambda_l}\\
&= \mathsf{Adv}^{\mathrm{CTI\text{-}sCPA}}_{\mathcal{E},\Lambda} &+ \mathsf{Adv}^{\mathrm{ERR\text{–}CPA}}_{\mathcal{E},\Lambda} &+ \mathsf{Adv}^{\mathrm{CTI\text{–}CPA}}_{\mathcal{E},\Lambda}\\
&\le 2\cdot\mathsf{Adv}^{\mathrm{CTI\text{–}sCPA}}_{\mathcal{E},\Lambda} &+ \mathsf{Adv}^{\mathrm{ERR\text{–}CPA}}_{\mathcal{E},\Lambda}
\end{aligned}$$

□

3 Comparison of Recent AE Notions

Three recent papers introduced strengthened AE notions to capture distinguishable decryption failures [19], releasing unverified plaintext [3], and "robust" authenticated encryption [29]. In every case the encryption oracle can be cast as

$$\mathrm{E}\colon \mathsf{K}\times\mathsf{N}\times\mathsf{A}\times\mathsf{M}\to\mathsf{C}$$

but their respective authors make slightly different definitional choices depending on which aspect of the implementation they had in mind when developing the notion. The main differences are how decryption and its leakage are defined, when a ciphertext is considered valid, and what security to aim for. A quick comparison of the syntactical choices is provided in Table 1.

In this section we will show how these three notions relate our framework. With the appropriate syntactical modifications, it turns out that each of these three notions can be cast into our framework, allowing us to conclude that the respective security notions are all broadly equivalent to our more general SAE notion. As an obvious corollary, the three existing notions turn out to be less radically different than initially suggested.

3.1 Distinguishable Decryption Failures (BDPS, [19])

The accepted notions of encryption typically only allow decryption to fail in one way, captured by the single error symbol $\perp$ [10, 33, 51]. Yet several provably LOR–CCA secure schemes have succumbed to practical attacks as a result of different decryption failures being distinguishable, both in the public key and symmetric settings [17, 57]. Boldyreva et al. [19] initiated a systematic study of the effects of symmetric schemes with multiple decryption errors. They emphasised probabilistic and stateful schemes, omitting a more modern nonce-based treatment. Below we describe the nonce-based analogues of their syntax and security notions.

A nonce-based, multi-error AE scheme a la BDPS, is a pair (E,D),

$$\begin{aligned}
\mathrm{E} &: \mathsf{K}\times\mathsf{N}\times\mathsf{A}\times\mathsf{M}\to\mathsf{C}\\
\mathrm{D} &: \mathsf{K}\times\mathsf{N}\times\mathsf{A}\times\mathsf{C}\to\mathsf{M}\cup\mathsf{L}
\end{aligned}$$

	SAE (Section 2.5)		BDPS [19]	RUP [3]		RAE [29]
	$\mathcal{D}_k$	Λ_k	D_k	D_k	V_k	Π^{-1}
Valid Ciphertext, $C = \mathcal{E}_k(M)$	$M \in \mathsf{M}$	$\top$	$M \in \mathsf{M}$	$M \in \mathsf{M}$	$\top$	$M \in \mathsf{M}$
Invalid Message, $c \in \mathsf{C} \setminus \mathsf{im}(\mathcal{E}_k)$	$\perp$	$\perp_i \in \mathsf{L}$	$\perp_i \in \mathsf{L}$	$\perp_i \in \mathsf{L} = \mathsf{M}$	$\perp$	$\perp_i \in \mathsf{L} = \mathsf{M}$, $\lvert\perp_i\rvert \neq \lvert c\rvert - \tau$

Table 1: Syntax for decryption leakage chosen by this and three previous works.

Our notion	IND–CPA	CTI–CPA	CTI–sCPA	IND–sCCA	IND–sPAS
Simplified bitstring	10000	01100	01101	10011	10001
BDPS notion	IND$–CPA	INT–CTXT*	INT–CTXT	IND$–CCA	IND$–CVA
Reference (in [19])	Def. 5	Def. 7	Def. 7	Def 5.	Def. 5
Direct translation	10000	01110	01111	10011	10001
RUP notion	IND–CPA	INT–CTXT	INT–RUP		
Reference (in [3])	Def 1.	Def. 4	Def. 8		
Direct translation	10000	01100	01111		

Table 2: Notions from BDPS and RUP that directly translate into our framework.

satisfying the classical definition of correctness. The idea is that if decryption fails, it may output any error symbol from L. (BDPS stipulate finite L, but this restriction appears superfluous and we omit it.)

To cast a (nonce-based) BDPS scheme into our SAE syntax, we observe the obvious (invertible) mapping from a scheme (E, D) by setting $\mathcal{E}_k = \mathrm{E}_k$, $\mathcal{D}_k(C) = \mathrm{D}_k(C)$ whenever $\mathrm{D}_k(C) \in \mathsf{M}$ or otherwise $\perp$, and $\Lambda_k(C) = \mathrm{D}_k(C)$ whenever $\mathrm{D}_k(C) \in \mathsf{L}$ or otherwise $\top$.

Notions. BDPS define a number of notions, including confidentiality notions following both the LOR and probabilistic IND$ concepts. Once adapted to a nonce-based setting, several of their notions directly translate into our framework, as listed in Table 2. Additionally, BDPS define *error invariance* [19, Def. 8], which (roughly) says that it should be hard for an adversary with access to honest encryption and decryption oracles to achieve any leakage other than a particular value. This notion, INV–ERR, implies that an adversary cannot learn anything from decryption leakage and can be thought of as a special case of ERR–CCA since the simulator need just return this common value.

Error invariance is strictly stronger than leakage simulatability: To see this, consider a scheme that is INV–ERR, then there is a simple variant that upon triggering an error returns $\perp_0$ or $\perp_1$ depending on the first ciphertext bit (instead of always returning the same $\perp$). Such a scheme is still ERR–CCA but no longer INV–ERR. We consider this a weakness of error invariance, rather than a benefit: effectively the definition has become so restrictive that it, in a roundabout way, again rules out multiple decryption errors.

The strongest goal defined by BDPS is IND$–CCA3 [19, Def. 19], which incorporates multiple errors to the classical authenticated encryption notion. It is characterised by two oracles: an adversary has to distinguish between $(\mathrm{E}_k, \mathrm{D}_k)$ (real) and $(\$, \perp)$ (ideal), where the error $\perp$ is a parameter of the notion. Thus, despite the desire to capture multiple errors, in the ideal case the adversary is still only presented with a single error symbol. This curious artefact results from using INV–ERR rather than ERR–CCA to characterise "acceptable" leakage. Consequently, the reference definition that does not model the real-world problem satisfactorily; for instance it fails to capture the release of unverified plaintext.

Implications and separations. BDPS provide several implications and separations between their notions. Although originally stated and proven for probabilistic and stateful schemes, the results easily carry over to a nonce-based setting. Using our naming convention, BDPS show that IND–sPAS + CTI–sCPA $\implies$ IND–sCCA, yet IND–sPAS + CTI–CPA $\not\Longrightarrow$ IND–sCCA. This immediately implies a separation between CTI–sCPA and CTI–CPA. They also prove that AE and INV–ERR jointly are equivalent to their IND$–CCA3 notion [19, Theorem 20], which itself implies IND–sPAS and CTI–sCPA.

Since INV–ERR implies ERR–CCA, this means IND$–CCA3 implies SAE. Moreover, the separation between ERR–CCA and INV–ERR carries over when comparing IND$–CCA3 and SAE. For completeness, we give the following theorem, which is a direct result of combining BDPS's Theorem 20 (after incorporating nonces) with the observation that INV–ERR is more restrictive than ERR–CCA, then applying Theorem 6:3.

Theorem 7. *The IND$–CCA3 notion of BDPS implies SAE, differing only with respect to the (overly restrictive) notion of error invariance. That is,*

$$\text{IND\$–CCA3} \iff \text{AE + INV–ERR} \implies \text{AE + ERR–CCA} \iff \text{SAE}.$$

3.2 Releasing Unverified Plaintext (RUP, [3])

Andreeva et al. [3] set out to model decryption more accurately for schemes that calculate a candidate plaintext before confirming its validity. In practice, such a candidate plaintext often becomes available (including to an adversary), even if validation fails. Examples include all schemes that need to decrypt before integrity can be checked (covering MAC-then-encrypt, MAC-and-encrypt, and encode-then-encipher) as well as schemes sporting online decryption (for instance single-pass CBC-then-MAC decryption). Andreeva et al. provide a large number of new definitions, covering security under decryption-leakage for both confidentiality and integrity.

Syntactical differences. The RUP paper models decryption using a decryption oracle D that *always* prepares a purported plaintext and a verification oracle V that determines whether the ciphertext was valid or not. The idea is that in honest implementations the output of D is only released if verification passes, yet security should hold even if the output of D is leaked before verification takes place (corresponding to "releasing unverified plaintext").

The RUP framework includes an explicit tag T as output of encryption (and input to decryption). However the tag and ciphertext terms are always used together, allowing us to consider the ciphertext as (C, T) instead. This pair can be injectively mapped into C, e.g. by $C' := C||T$ if the stretch is fixed. With this modification in mind, the RUP algorithms D and V satisfy

$$\mathrm{D} : \mathsf{K} \times \mathsf{N} \times \mathsf{A} \times \mathsf{C} \to \mathsf{M} = \mathsf{L}$$
$$\mathrm{V} : \mathsf{K} \times \mathsf{N} \times \mathsf{A} \times \mathsf{C} \to \{\top, \bot\} .$$

When called with a valid ciphertext, D_k returns the plaintext, and V_k returns $\top$. Conversely, when called with an invalid ciphertext, D_k will return some leakage information (nominally, the eponymous "unverified plaintext") and V_k will return $\bot$.

By changing perspective, we can cast a RUP scheme into the SAE framework: let $\mathcal{D}_k(C) = \mathrm{D}_k(C)$ if $\mathrm{V}_k(C) = \top$ (otherwise $\mathcal{D}_k(C) = \bot$) and $\Lambda_k(C) = \mathrm{D}_k(C)$ whenever $\mathrm{V}_k(C) = \bot$ (otherwise $\Lambda_k(C) = \top$). Then, $(\mathcal{E}, \mathcal{D}, \Lambda)$ is an SAE scheme (where $\mathcal{E} = \mathrm{E}$), with leakage space $\mathsf{L} = \mathsf{M}$ (see also Table 2). For the remainder of this section, we will abuse the Δ notation defined previously to describe access to three oracles matching the syntax of $\mathrm{E}_k, \mathrm{D}_k$ and V_k respectively.

Security notions. Andreeva et al. refer to the traditional "encryption-only" notions of confidentiality and integrity under their modern names IND–CPA and INT–CTXT (our CTI–CPA). When decryption comes into play, a large number of new notions are suggested, typically defined in terms of adversarial access to their D_k and V_k oracles.

For integrity, the adversary is given full access to all three honest oracles ($\mathcal{E}_k, \mathrm{D}_k$, and V_k), and challenged to make a forgery, a notion dubbed INT–RUP for integrity under release of unverified plaintext. Since access to both D_k and V_k is equivalent to access to (our) $\mathcal{D}_k$ and Λ_k, INT–RUP directly translates into our framework: it corresponds to CTI-sCCA (itself equivalent to CTI–sCPA). This makes Andreeva et al.'s INT–RUP equivalent to BDPS's INT–CTXT notion. These observations are presented in Table 2.

RUP Privacy Notion	IND–CCA	IND–CCA'	PA1	PA2	DI
Reference (in [3])	Def. 2	Def. 3	Def. 5	Def. 6	Def. 7
Game	$\Delta^{\mathrm{E}_k,\mathrm{D}_k,\emptyset}_{\$,\mathrm{D}_k,\emptyset}$	$\Delta^{\mathrm{E}_k,\mathrm{D}_k,\emptyset}_{\$,\mathrm{D}_k,\emptyset}$	$\Delta^{\mathrm{E}_k,\mathrm{D}_k,\emptyset}_{\mathrm{E}_k,\mathsf{S},\emptyset}$	$\Delta^{\mathrm{E}_k,\mathrm{D}_k,\emptyset}_{\mathrm{E}_k,\mathsf{S},\emptyset}$	$\Delta^{\mathrm{E}_k,\mathrm{D}_k,\emptyset}_{\mathrm{E}_k,\mathrm{D}_l,\emptyset}$
Comments	Full domain separation required		S is a stateful simulator with access to query history of E_k	S is a stateful simulator	$l \leftarrow\!\!\$\ \mathsf{K}$, named for "Decryption Independence"

Table 3: RUP confidentiality notions that do not readily translate into our framework. In each case, domain separation requires we do not forward queries from Enc to Dec to prevent trivial wins.

For most of the RUP confidentiality notions the verification oracle is missing. This makes translation into our syntax cumbersome as any direct method would implicitly provide access to V_k functionality. Table 3 lists five RUP confidentiality notions in their original syntax and nomenclature.

As far as combined notions are concerned, RUP defines the overall goal as achieving the triple of INT–RUP, DI and IND–CPA. They also define AE–RUP2, which aims to fulfil a similar purpose to ERR–CCA.

Implications and separations.. Andreeva et al. show that their PA2 and DI are equivalent [3, Theorems 8 and 9] and imply PA1 [3, Theorem 1]. Moreover, when combined with IND–CPA, PA2 provides a meaningful increase in security [3, Theorems 2 and 3], whereas PA1 does not [3, Theorems 4 and 5]. They provide an alternative proof that CTI–sCPA is strictly stronger than CTI–CPA [3, Theorem 10].

Critique. The RUP model restricts any decryption leakage to the message space. This is unnecessarily restrictive: it does not directly cover multiple decryption errors; moreover a scheme may conceivably leak some internal variable (say a buffer) that is not in the message space.

For the myriad of RUP's confidentiality notions, an adversary is—for whatever reason—not provided with a verification oracle. For instance, RUP provides two versions of IND–CCA security. Each gives the adversary access to a challenge Enc oracle and the honest decryption/leakage function D_k, but neither provides access to V_k, with the difference between them being whether the adversary may forward queries from D_k to E_k or not.

This restriction is similar to those we discuss in Appendix B and raises the question whether either of RUP's IND–CCA notions imply traditional LOR–CCA once the leakage is ignored. If the scheme is tidy, the RUP decryption and encryption oracles together suffice to implement the decrypt-verify oracle, and this can be used to show that IND–CCA' security even implies AE security. However, RUP's IND–CCA prevents this inter-use of oracles, and in fact it appears that it does not imply traditional LOR–CCA. These claims are stated in Lemma 8 and Lemma 9, proofs of which can be found in Appendix C.

Lemma 8. *RUP's* IND–CCA' *implies* AE.

Lemma 9. *RUP's* IND–CCA *does not imply either our* IND–CCA *or traditional* LOR–CCA

Authenticated encryption definition. Andreeva et al. suggest that an authenticated encryption should meet the combined goals of IND–CPA and PA for confidentiality, and INT–RUP (our CTI-sCPA) for integrity [3, §8].

Having to satisfy three separate notions may appear needlessly complicated and lacks the elegance a single notion can provide. We propose *RUPAE* as a natural and neater way of defining Andreeva et al.'s final objective, where we use (the equivalent) DI instead of the less direct PA:

$$\mathsf{Adv}^{\mathrm{RUPAE}}_{\mathcal{E},\Lambda} := \Delta^{\mathrm{E}_k,\mathrm{D}_k,\mathrm{V}_k}_{\$,\mathrm{D}_l,\perp}$$

This goal may originally have been envisaged by the authors, yet it was not explicitly alluded to (let alone defined). Providing a single succinct security goal is only worthwhile if it properly captures the the compound notions, which we show in Theorem 10. The proof is intuitive, based around liberal use of the triangle inequality.

Theorem 10. *The single term RUPAE notion is equivalent to the triple of goals originally proposed. That is,*

$$\text{RUPAE} \iff \text{CTI–sCPA + DI + IND–CPA} \iff \text{INT–RUP + PA + IND–CPA}.$$

Proof. The second equivalence is simply a renaming, so we must just prove the first.

$$\begin{aligned}\mathsf{Adv}^{\text{RUPAE}}_{\mathcal{E},\Lambda} = \Delta^{\mathrm{E}_k,\mathrm{D}_k,\mathrm{V}_k}_{\$\,,\mathrm{D}_l\,,\,\perp} &\le \Delta^{\mathrm{E}_k,\mathrm{D}_k,\mathrm{V}_k}_{\mathrm{E}_k,\mathrm{D}_k,\,\perp} + \Delta^{\mathrm{E}_k,\mathrm{D}_k,\perp}_{\mathrm{E}_k,\mathrm{D}_l\,,\perp} + \Delta^{\mathrm{E}_k,\mathrm{D}_l,\perp}_{\$\,,\mathrm{D}_l,\perp}\\ &= \Delta^{\mathrm{E}_k,\mathrm{D}_k,\mathrm{V}_k}_{\mathrm{E}_k,\mathrm{D}_k,\,\perp} + \Delta^{\mathrm{E}_k,\mathrm{D}_k,\emptyset}_{\mathrm{E}_k,\mathrm{D}_l\,,\emptyset} + \Delta^{\mathrm{E}_k,\emptyset,\emptyset}_{\$\,,\emptyset,\emptyset}\\ &= \mathsf{Adv}^{\text{CTI–sCPA}}_{\mathcal{E},\Lambda} + \mathsf{Adv}^{\text{DI}}_{\mathcal{E},\Lambda} + \mathsf{Adv}^{\text{IND–CPA}}_{\mathcal{E},\Lambda}\end{aligned}$$

Thus if these three advantages are small so is the RUPAE advantage, and the scheme is secure. Conversely,

$$\begin{aligned}\mathsf{Adv}^{\text{IND–CPA}}_{\mathcal{E},\Lambda} = \Delta^{\mathrm{E}_k,\,\emptyset\,,\emptyset}_{\$\,,\,\emptyset\,,\emptyset} &\le \Delta^{\mathrm{E}_k,\mathrm{D}_k,\mathrm{V}_k}_{\$\,,\mathrm{D}_l\,,\,\perp} &&&&\le \mathsf{Adv}^{\text{RUPAE}}_{\mathcal{E},\Lambda}\\ \mathsf{Adv}^{\text{DI}}_{\mathcal{E},\Lambda} = \Delta^{\mathrm{E}_k,\mathrm{D}_k,\emptyset}_{\mathrm{E}_k,\mathrm{D}_l\,,\emptyset} &\le \Delta^{\mathrm{E}_k,\mathrm{D}_k,\mathrm{V}_k}_{\$\,,\mathrm{D}_l,\,\perp} + \Delta^{\$\,,\mathrm{D}_l,\perp}_{\mathrm{E}_k,\mathrm{D}_l,\perp}\\ &\le \mathsf{Adv}^{\text{RUPAE}}_{\mathcal{E},\Lambda} + \mathsf{Adv}^{\text{IND–CPA}}_{\mathcal{E},\Lambda} &&&&\le 2\cdot\mathsf{Adv}^{\text{RUPAE}}_{\mathcal{E},\Lambda}\\ \mathsf{Adv}^{\text{CTI–sCPA}}_{\mathcal{E},\Lambda} = \Delta^{\mathrm{E}_k,\mathrm{D}_k,\mathrm{V}_k}_{\mathrm{E}_k,\mathrm{D}_k,\,\perp} &\le \Delta^{\mathrm{E}_k,\mathrm{D}_k,\mathrm{V}_k}_{\$\,,\mathrm{D}_l,\,\perp} + \Delta^{\$\,,\mathrm{D}_l,\perp}_{\mathrm{E}_k,\mathrm{D}_l,\perp} + \Delta^{\mathrm{E}_k,\mathrm{D}_l,\perp}_{\mathrm{E}_k,\mathrm{D}_k,\perp}\\ &\le \mathsf{Adv}^{\text{RUPAE}}_{\mathcal{E},\Lambda} + \mathsf{Adv}^{\text{IND–CPA}}_{\mathcal{E},\Lambda} + \mathsf{Adv}^{\text{PA}}_{\mathcal{E},\Lambda} &&&&\le 4\cdot\mathsf{Adv}^{\text{RUPAE}}_{\mathcal{E},\Lambda}\end{aligned}$$

Thus the opposite relation also holds, and if the RUPAE advantage is small so are the CTI–sCPA, DI and IND–CPA advantages. □

To relate this to our other notions, we provide the following observation:

Lemma 11. $\text{CTI–sCPA + ERR–CPA} \iff \text{CTI–sCPA + DI}$

Proof. We must show that given CTI–sCPA, ERR–CPA and DI coincide.

We apply conditional probability to expand out DI, where throughout $k, l \leftarrow\!\!\$\ \mathsf{K}$. Let F be the event that $\mathcal{A}$ makes a call to his decryption oracle with a valid ciphertext (i.e. he creates and uses a forgery). If $\neg\mathrm{F}$, $\mathcal{A}$ has not been able to create a forgery and so every call to the D oracle returns only values from the leakage function. Therefore, as long as $\neg\mathrm{F}$, ERR–CPA and DI define the same game. Since the probability $\mathbb{P}[\mathrm{F}]$ is precisely the probability an adversary wins the CTI-sCPA game,

$$\left|\mathsf{Adv}^{\text{ERR–CPA}}_{\mathcal{E},\Lambda}(\mathcal{A}) - \mathsf{Adv}^{\text{DI}}_{\mathcal{E},\Lambda}(\mathcal{A})\right| \le \mathbb{P}[\mathrm{F}] \le \mathsf{Adv}^{\text{CTI-sCPA}}_{\mathcal{E},\Lambda}(\mathcal{A})\,.$$

So, as long as the CTI-sCPA advantage is small, ERR-CPA and DI are equivalent. Thus CTI–sCPA + ERR–CPA $\iff$ CTI–sCPA + DI. □

Finally then, we have the reassuring result that security within the RUP framework coincides with our more general definition. To prove it, one simply chains Theorem 6, Lemma 11, and Theorem 10 (in that order).

Corollary 12. *RUPAE security is equivalent to SAE security.*

3.3 Robust Authenticated Encryption (RAE, [29])

The proposal of Hoang et al. [29, §3], Robust Authenticated Encryption (RAE), has robustness against inadvertent leakage of unverified plaintext as one of it goals. A notable difference compared to other notions of AE is that RAE targets schemes where for *each* message the intended level of integrity is explicitly specified by the user. To this end, both encryption and decryption algorithms are provided with an additional input, called the stretch parameter τ, leading to the syntax:

$$\begin{aligned}\mathrm{E} &: \mathsf{K}\times\mathsf{N}\times\mathsf{A}\times\mathbb{N}\times\mathsf{M}\to\mathsf{C}\\ \mathrm{D} &: \mathsf{K}\times\mathsf{N}\times\mathsf{A}\times\mathbb{N}\times\mathsf{C}\to\mathsf{M}\,.\end{aligned}$$

Thus, encryption calls are of the form $C = \mathrm{E}_k^{N,A,\tau}(M)$: they accept a nonce N, some associated data A, the stretch parameter τ and a message M, and output some ciphertext C. There is a requirement that τ is indeed the stretch, namely that $|C| = |M| + \tau$. Decryption calls take a similar format, and are allowed to "leak" a string *not* of the correct length when queried with invalid inputs. This length restriction on the leakage implies that valid ciphertexts can easily be determined from their length. A ciphertext is deemed valid iff $|\mathrm{D}_k^{N,A,\tau}(C)| = |C| - \tau$, moreover if $M = \mathrm{D}_k^{N,A,\tau}(C)$ with $|M| = |C| - \tau$ then it follows that $\mathrm{E}_k^{N,A,\tau}(M) = C$.

The security game. The RAE security game aims to describe the *best possible* security for an object with the given syntax. Comparison to ideal objects is not new: it is the standard notion for blockciphers (namely a strong pseudorandom permutation [39]) and has appeared previously as an alternative for deterministic authenticated encryption (in the form of strong pseudorandom injections [51]).

For a given stretch τ, the ideal object is a random element of $\mathrm{Inj}(\tau)$, the set of all injective functions whose outputs are always τ bits longer than their input. For $\tau > 0$, strings outside the range of $\pi \in \mathrm{Inj}(\tau)$ are deemed incorrect ciphertexts and π cannot be inverted mathematically on these elements. Since decryption may leak on these incorrect ciphertexts, returning $\perp$ in that case is no longer an option either.

Hoang et al. solve this problem by introducing a stateful simulator S and its filtered variant S_π to answer decryption queries in the ideal world. Here S_π is the simulator S filtered by (the complement of) the range of π; it exhibits the following behaviour. If a decryption query is valid (i.e. $C \in \mathsf{Image}(\pi)$), then the simulator S_π returns the unique preimage M. (That membership of $\mathsf{Image}(\pi)$ cannot be determined efficiently is not an issue.) Otherwise, the ciphertext is invalid ($C \notin \mathsf{Image}(\pi)$) and S_π calls S, which must simulate the decryption oracle *without* access to the injections $\pi_{\cdot,\cdot,\cdot}$. The simulator S will output a bitstring of any length other than $|C| - \tau$, which the S_π oracle will simply forward as its own output.

Security is defined relative to this simulator S and in terms of distinguishing between two worlds:

$$\mathsf{Adv}^{\mathrm{RAE}}_{\Pi,\mathsf{S}} := \mathbb{P}\left[k \leftarrow\!\!\$\, \mathsf{K} \colon \mathcal{A}^{\mathrm{E}_k,\mathrm{D}_k} \to 1\right] - \mathbb{P}\left[\pi_{N,A,\tau} \leftarrow\!\!\$\, \mathrm{Inj}(\tau) \colon \mathcal{A}^{\pi,\mathsf{S}_\pi} \to 1\right] .$$

The second probability above contains a slight abuse of notation, as the injections $\pi_{N,A,\tau}$ are tweaked by the nonce, associated data, and stretch τ. A fully formalised, code-based description can be found in the original paper [29, World $\mathbf{RAE}_{\Pi,S}$ in Fig. 2].

Ideal versus attainable. Following Rogaway's definitional papers [47, 48, 51], most recent symmetric results have been given in terms of indistinguishability from the ideal world $(\$, \perp)$: an ideal encryption oracle that outputs random bits and an ideal decryption oracle that never accepts. Hoang et al. instead opt for a reference world that corresponds to the "best achievable", a contrast they emphasize: "Before, AE–quality was always measured with respect to an aspirational goal; now we're suggesting to employ an achievable one." [29, §1:Discussion]. So, one of the big differences between RAE and the surrounding literature is that RAE compares a scheme with the *achievable* world, rather than the *ideal* one.

One can see the resulting security notions as (ever more complicated) extensions of the pseudorandom permutation notion typical for blockcipher security. This immediately reveals that to some extent, this choice is one of fixing abstraction boundaries. When purely studying how to transform one primitive to another, it makes sense to use the achievable world as a benchmark, since the results may be tighter. Yet, for an overall security definition that is both robust and meaningful the reference world should be ideal, such that the overall bound really does represent the real-world security of the scheme. When non-experts use a robust AE scheme in larger designs, there should be as few implementation and configuration pitfalls as possible, and a small adversarial advantage should imply security as intuitively understood.

Regarding user-defined stretch. The variable, user-defined stretch sets RAE apart from other notions discussed so far. Hoang et al. insist that all values of stretch, including $\tau = 0$, should be allowed for a

scheme to be called RAE [29, footnote 9]. Consequently, RAE provides a continuum from authenticated encryption to unauthenticated enciphering. As Hoang et al. hasten to add, a small stretch makes forging trivial, rendering a good (generic) upper bound on the CTI–PAS advantage impossible.

While we consider RAE a clever and potent *primitive* spanning from ciphers to encryption, we find the *terminology* "robust authenticated" encryption rather worrying. Robustness characterises the ability of a construct to be pushed right to the edge of its intended use case (and possibly beyond). However, robustness resides in the implementation and deployment, not the primitive at hand: in App. E we demonstrate a natural yet insecure implementation of supposedly RAE-secure primitive AEZ. What is worse, even if an RAE primitive is properly implemented, exposing its full API to end users is an invitation to end up with unauthenticated encryption, contrary to the name RAE suggesting a strengthened version of authenticated encryption.

A crucial question when dealing with RAE is therefore who is the intended user who will choose the stretch. If these are cryptographers designing further primitives (such as SAE) and protocols (such as secure channels or disk encryption), then the variable stretch introduces expressiveness that might otherwise be missing. It is then up to the primitive and protocol designers to guard users of these constructs built on top of RAE from 'bad' stretches. Inevitably, the larger the user base, the less appreciation for configuration subtleties.

Fixing the stretch. There is no intrinsic reason to bar a scheme from restricting which values of τ it supports; the RAE security definition still makes perfect sense for length-regular schemes, where the stretch is no longer user-defined (but instead depends only on the length of the input message). To ease comparison with previous security notions, we will depart from Hoang et al.'s insistence on allowing small stretch. Henceforth we will restrict attention to schemes with fixed stretch $\tau \in \mathbb{N}$ and refer to this restricted notion as RAE[τ].

The mapping that takes an RAE[τ] scheme to an SAE scheme is rather intuitive, and analogous to that used in Section 3.2. Explicitly, let (E, D) be an RAE[τ] scheme, and (inspired by RUP) let V_k be the associated validity function, where $\mathrm{V}_k^{N,A}(C) = \top \iff |\mathrm{D}_k^{N,A,\tau}(C)| - |C| = \tau$. Then $(\mathcal{E}, \mathcal{D}, \Lambda)$ is an SAE scheme, where $\mathcal{E}_k^{N,A}(M) := \mathrm{E}_k^{N,A,\tau}(M)$ and

$$\mathcal{D}_k^{N,A}(C) := \begin{cases} \mathrm{D}_k^{N,A,\tau}(C) & \text{if } \mathrm{V}_k^{N,A}(C) = \top \\ \bot & \text{if } \mathrm{V}_k^{N,A}(C) \neq \top \end{cases}, \qquad \Lambda_k^{N,A}(C) := \begin{cases} \top & \text{if } \mathrm{V}_k^{N,A}(C) = \top \\ \mathrm{D}_k^{N,A,\tau}(C) & \text{if } \mathrm{V}_k^{N,A}(C) \neq \top \end{cases}$$

RAE[τ] versus SAE. Having applied the transform (which has no bearing on security), it is not surprising to find RAE[τ] and SAE security essentially coincide, with the only complication a generic term, reflecting the difference between ideal and best possible security. In order to provide an explicit relationship between the two notions, we first present a simulator-free version of RAE[τ] security (an analogue of Lemma 4).

Lemma 13. *Let Λ be the simulator that initializes (once) by sampling a key $l \leftarrow\!\!\$\ \mathsf{K}$, then evaluates Λ_l for each query. Then for any simulator S it holds that $\mathsf{Adv}^{\mathrm{RAE}[\tau]}_{\Pi,\Lambda}(\mathcal{A}) \leq 2 \cdot \mathsf{Adv}^{\mathrm{RAE}[\tau]}_{\Pi,\mathsf{S}}(\mathcal{A})$.*

Proof. Consider the image of the ideal world under the map from the RAE[τ] context to the SAE framework. Idealised encryption again forwards calls π, idealised decryption now forwards all calls π^{-1}, and the image of leakage is now simply a call to S. Thus being secure under RAE[τ] is equivalent to saying an adversary cannot distinguish between $(\mathcal{E}_k, \mathcal{D}_k, \Lambda_k)$ and $(\pi, \pi^{-1}, \mathsf{S})$. So,

$$\begin{aligned} \mathsf{Adv}^{\mathrm{RAE}[\tau]}_{\Pi,\Lambda}(\mathcal{A}) &= \Delta^{\mathcal{E}_k, \mathcal{D}_k, \Lambda_k}_{\pi, \pi^{-1}, \Lambda_l}(\mathcal{A}) \\ &\leq \Delta^{\mathcal{E}_k, \mathcal{D}_k, \Lambda_k}_{\pi, \pi^{-1}, \mathsf{S}}(\mathcal{A}) + \Delta^{\pi, \pi^{-1}, \mathsf{S}}_{\pi, \pi^{-1}, \Lambda_l}(\mathcal{A}) \\ &\leq 2 \cdot \mathsf{Adv}^{\mathrm{RAE}[\tau]}_{\Pi,\mathsf{S}}(\mathcal{A}) \end{aligned}$$

Thus whenever there is a simulator such that the scheme is RAE[τ] secure, Λ describes one also. □

Theorem 14. *RAE[τ] and SAE security are equivalent. Explicitly, for an adversary $\mathcal{A}$ making at most q queries, and using a repeated nonces r times,*

$$\left|\mathsf{Adv}^{\mathrm{RAE}[\tau]}_{\Pi,\Lambda}(\mathcal{A}) - \mathsf{Adv}^{\mathrm{SAE}}_{\mathcal{E},\Lambda}(\mathcal{A})\right| \leq \tfrac{q}{2^{\tau-1}} + \tfrac{r^2+r}{2^{\tau+m+1}} \ .$$

Proof. As discussed in the previous lemma, we may port the RAE[τ] ideal world into our framework, and assume that the simulator is Λ. So,

$$\begin{aligned}\left|\mathsf{Adv}^{\mathrm{RAE}[\tau]}_{\Pi,\Lambda_l} - \mathsf{Adv}^{\mathrm{SAE}}_{\mathcal{E},\Lambda}\right| &= \left|\Delta^{\mathcal{E}_k,\,\mathcal{D}_k\,,\Lambda_k}_{\pi\,,\pi^{-1},\Lambda_l} - \Delta^{\mathcal{E}_k,\mathcal{D}_k,\Lambda_k}_{\$\,,\,\perp\,,\Lambda_k}\right| \\ &= \Delta^{\$,\ \perp\ ,\Lambda_k}_{\pi,\pi^{-1},\Lambda_l} \\ &\leq \Delta^{\$,\perp,\Lambda_k}_{\$,\perp,\Lambda_l} + \Delta^{\$,\ \perp\ ,\Lambda_l}_{\pi,\pi^{-1},\Lambda_l}\end{aligned}$$

The first of these is the ERR–PAS game, which is perfectly secure. So, the only difference between the schemes is the second term, characterising the choice of ideal world, which is precisely the difference between the PRI and MRAE security games (since the leakage is miskeyed and thus irrelevant). To reach an explicit result, we use the concrete security result of Hoang et al. [29, Theorem 1] to bound the PRI–MRAE switch. □

4 Conclusion

This paper investigated the effect of deterministic decryption leakage on provably secure authenticated encryption. Having clarified the formalisation of this, we defined the appropriate notions, and compared them, both to each other and to some previous works. With hindsight then, the focal point of this paper is the Cube of notions and relations in Figure 4, which provides a visualisation of the modern nonce-based world.

We first formalised what it meant to be a Subtle Authenticated Encryption (SAE) scheme (Section 2), building on previous notions for tidy nonce-based AEAD. This was followed by defining the associated security notions, where we provide a systematic naming scheme, that is both an intuitive and descriptive, separating the adversary's goal from their powers. For goals, we use IND (for INDistinguishability) to measure a schemes confidentiality, CTI (for CipherText Integrity) to measure authenticity and introduce ERR (for ERRor simulatability) to describe how significant the leakage is. Our powers follow conventional naming, with CPA,CCA and PAS taking their traditional meanings, while CDA corresponds to a chosen decryption attack and the subtle variants (e.g. sCPA) are those where the adversary may also observe leakage. In particular, an SAE scheme is secure if the adversarial advantage in the AE-sCCA game is sufficiently small. Our study then moved on to demonstrating equivalences, separations and relations between the different notions. This culminates in the relationships described in the cube (Figure 4), with one key result being that ERR–CCA precisely describes the difference between AE security in the absence of leakage and full SAE security.

With the benchmark for security set, we compare it with previous attempted formalisations (Section 3). Of these, BDPS [19] provides the most generalised syntax, although the condition that errors be invariant and the (seemingly unnecessary) requirement of a finite error space limit the applicability of their results. RUP [3] presents the material in a very practical way, with definitions and models that clearly describe how decrypt-then-verify schemes behave, but in doing so yield a scheme that does not readily generalise to handling alternative leakage sources. RAE [29] on the other hand defines a goal that, at first glance, appears to be the strongest of them all, but upon further inspection is rather more nuanced. After minor changes, the key definitions of all three papers turn out to be equivalent to our intuitive SAE notion, meaning the three papers have more in common than perhaps originally recognised by their authors.

To compliment the work, we present an overview of the literature on which this work is built (Appendix A), and observe that this is slightly more nuanced than one might believe. As an example, we investigate the case of nonce-based IND–CCA, finding seemingly minor definitional choices yield very

different notions (Appendix B), and demonstrating how important it is that we treat leakage as a property of the actual implementation not the abstract functionality (Appendix E).

In this paper we focused on how one should characterise security in the presence of deterministic decryption leakage, sidestepping the related question of how one should achieve security within such a framework. It would be interesting for future works to describe classes of scheme that can (or cannot) achieve subtle security to a meaningful extent. Currently, very few schemes meet these strong definitions of security. RUP demonstrates that a number of popular schemes cannot possibly do so by providing attacks, and goes on to prove that certain schemes achieve a weaker version of security (where leaked plaintext can only be shown harmless by a simulator with access to the query history of the encryption oracle). However, at the time of writing, the only method known for achieving SAE security is through the encode-then-encipher paradigm (as proven in RAE).

Acknowledgements. We thank Dan Martin and Elisabeth Oswald for fruitful discussions regarding leakage-resilience and the anonymous referees of the IMA International Conference on Cryptography and Coding 2015 for their constructive feedback.

This work was conducted whilst Guy Barwell was a PhD student at the University of Bristol, supported by an EPSRC grant.

References

[1] Abed, F., Fluhrer, S.R., Forler, C., List, E., Lucks, S., McGrew, D.A., Wenzel, J.: Pipelineable on-line encryption. In: Cid and Rechberger [20], pp. 205–223; Cited on page 15.

[2] Abed, F., Forler, C., List, E., Lucks, S., Wenzel, J.: Don't Panic! The Cryptographers' Guide to Robust Authenticated (On-line) Encryption. Comments to CAESAR mailing list (2015); Cited on pages 8 and 34.

[3] Andreeva, E., Bogdanov, A., Luykx, A., Mennink, B., Mouha, N., Yasuda, K.: How to securely release unverified plaintext in authenticated encryption. In: Sarkar, P., Iwata, T. (eds.) ASIACRYPT 2014, Part I. LNCS, vol. 8873, pp. 105–125. Springer, Heidelberg (Dec 2014); Cited on pages 2, 3, 5, 7, 16, 19, 20, 21, 22, 26, 33, 34, and 36.

[4] Andreeva, E., Bogdanov, A., Luykx, A., Mennink, B., Tischhauser, E., Yasuda, K.: Parallelizable and authenticated online ciphers. In: Sako, K., Sarkar, P. (eds.) ASIACRYPT 2013, Part I. LNCS, vol. 8269, pp. 424–443. Springer, Heidelberg (Dec 2013); Cited on page 14.

[5] Atluri, V. (ed.): ACM CCS 02. ACM Press (Nov 2002); Cited on pages 28 and 29.

[6] Barwell, G., Page, D., Stam, M.: Rogue decryption failures: Reconciling AE robustness notions. In: Groth, J. (ed.) 15th IMA International Conference on Cryptography and Coding. LNCS, vol. 9496, pp. 94–111. Springer, Heidelberg (Dec 2015); Cited on page 1.

[7] Bauer, A., Coron, J.S., Naccache, D., Tibouchi, M., Vergnaud, D.: On the broadcast and validity-checking security of PKCS#1 v1.5 encryption. In: Zhou, J., Yung, M. (eds.) ACNS 10. LNCS, vol. 6123, pp. 1–18. Springer, Heidelberg (Jun 2010); Cited on page 8.

[8] Bellare, M., Boldyreva, A., Knudsen, L.R., Namprempre, C.: Online ciphers and the hash-CBC construction. In: Kilian [35], pp. 292–309; Cited on page 32.

[9] Bellare, M., Brakerski, Z., Naor, M., Ristenpart, T., Segev, G., Shacham, H., Yilek, S.: Hedged public-key encryption: How to protect against bad randomness. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 232–249. Springer, Heidelberg (Dec 2009); Cited on page 8.

[10] Bellare, M., Desai, A., Jokipii, E., Rogaway, P.: A concrete security treatment of symmetric encryption. In: 38th FOCS. pp. 394–403. IEEE Computer Society Press (Oct 1997); Cited on pages 3, 10, 12, 19, 30, 35, and 36.

[11] Bellare, M., Goldreich, O., Mityagin, A.: The power of verification queries in message authentication and authenticated encryption. Cryptology ePrint Archive, Report 2004/309 (2004), http://eprint.iacr.org/2004/309; Cited on pages 15 and 31.

[12] Bellare, M., Hofheinz, D., Kiltz, E.: Subtleties in the definition of IND-CCA: When and how should challenge decryption be disallowed? Journal of Cryptology 28(1), 29–48 (Jan 2015); Cited on pages 9 and 37.

[13] Bellare, M., Kohno, T., Namprempre, C.: Authenticated encryption in SSH: Provably fixing the SSH binary packet protocol. In: Atluri [5], pp. 1–11; Cited on pages 31 and 33.

[14] Bellare, M., Namprempre, C.: Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. In: Okamoto [45], pp. 531–545; Cited on pages 12, 31, 36, and 39.

[15] Bellare, M., Rogaway, P.: Encode-then-encipher encryption: How to exploit nonces or redundancy in plaintexts for efficient cryptography. In: Okamoto [45], pp. 317–330; Cited on pages 31, 35, and 36.

[16] Bernstein, D.J.: CAESAR competition call (2013), http://competitions.cr.yp.to/caesar-call-3.html; Cited on page 3.

[17] Bleichenbacher, D.: Chosen ciphertext attacks against protocols based on the RSA encryption standard PKCS #1. In: Krawczyk, H. (ed.) CRYPTO'98. LNCS, vol. 1462, pp. 1–12. Springer, Heidelberg (Aug 1998); Cited on page 19.

[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. LNCS, vol. 7237, pp. 682–699. Springer, Heidelberg (Apr 2012); Cited on pages 33 and 34.

[19] Boldyreva, A., Degabriele, J.P., Paterson, K.G., Stam, M.: On symmetric encryption with distinguishable decryption failures. In: Moriai, S. (ed.) FSE 2013. LNCS, vol. 8424, pp. 367–390. Springer, Heidelberg (Mar 2014); Cited on pages 2, 3, 5, 8, 19, 20, 26, and 36.

[20] Cid, C., Rechberger, C. (eds.): FSE 2014, LNCS, vol. 8540. Springer, Heidelberg (Mar 2015); Cited on pages 28 and 29.

[21] Davies, G.T., Stam, M.: KDM security in the hybrid framework. In: Benaloh, J. (ed.) CT-RSA 2014. LNCS, vol. 8366, pp. 461–480. Springer, Heidelberg (Feb 2014); Cited on page 8.

[22] Dent, A.W.: A designer's guide to KEMs. In: Paterson, K.G. (ed.) 9th IMA International Conference on Cryptography and Coding. LNCS, vol. 2898, pp. 133–151. Springer, Heidelberg (Dec 2003); Cited on page 8.

[23] Dodis, Y., Pietrzak, K.: Leakage-resilient pseudorandom functions and side-channel attacks on Feistel networks. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 21–40. Springer, Heidelberg (Aug 2010); Cited on page 6.

[24] Dolev, D., Dwork, C., Naor, M.: Nonmalleable cryptography. SIAM Journal on Computing 30(2), 391–437 (2000) ; Cited on page 31.

[25] Dziembowski, S., Pietrzak, K.: Leakage-resilient cryptography. In: 49th FOCS. pp. 293–302. IEEE Computer Society Press (Oct 2008); Cited on page 5.

[26] Fischlin, M., Günther, F., Marson, G.A., Paterson, K.G.: Data is a stream: Security of stream-based channels. In: Gennaro, R., Robshaw, M.J.B. (eds.) CRYPTO 2015, Part II. LNCS, vol. 9216, pp. 545–564. Springer, Heidelberg (Aug 2015); Cited on page 34.

[27] Goldwasser, S., Micali, S.: Probabilistic encryption. Journal of Computer and System Sciences 28(2), 270–299 (1984); Cited on page 30.

[28] Hoang, V.T., Krovetz, T., Rogaway, P.: AEZ v4.1: Authenticated Encryption by Enciphering (2015), http://web.cs.ucdavis.edu/~rogaway/aez/; Cited on page 44.
[29] Hoang, V.T., Krovetz, T., Rogaway, P.: Robust authenticated-encryption AEZ and the problem that it solves. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015, Part I. LNCS, vol. 9056, pp. 15–44. Springer, Heidelberg (Apr 2015); Cited on pages 2, 3, 5, 8, 19, 20, 23, 24, 25, 26, 34, 44, and 45.
[30] Hoang, V.T., Reyhanitabar, R., Rogaway, P., Vizár, D.: Online authenticated-encryption and its nonce-reuse misuse-resistance. In: Gennaro, R., Robshaw, M.J.B. (eds.) CRYPTO 2015, Part I. LNCS, vol. 9215, pp. 493–517. Springer, Heidelberg (Aug 2015); Cited on page 8.
[31] Iwata, T., Minematsu, K., Guo, J., Morioka, S.: CLOC: Authenticated encryption for short input. In: Cid and Rechberger [20], pp. 149–167; Cited on pages 3, 7, 15, and 36.
[32] Katz, J., Lindell, Y.: Introduction to Modern Cryptography. Chapman & Hall/CRC (2008); Cited on pages 7 and 30.
[33] Katz, J., Yung, M.: Unforgeable encryption and chosen ciphertext secure modes of operation. In: Schneier, B. (ed.) FSE 2000. LNCS, vol. 1978, pp. 284–299. Springer, Heidelberg (Apr 2001); Cited on pages 19 and 31.
[34] Katz, J., Yung, M.: Characterization of security notions for probabilistic private-key encryption. Journal of Cryptology 19(1), 67–95 (Jan 2006); Cited on pages 37 and 38.
[35] Kilian, J. (ed.): CRYPTO 2001, LNCS, vol. 2139. Springer, Heidelberg (Aug 2001); Cited on pages 28 and 29.
[36] Krawczyk, H.: The order of encryption and authentication for protecting communications (or: How secure is SSL?). In: Kilian [35], pp. 310–331; Cited on page 37.
[37] Krovetz, T., Rogaway, P.: The software performance of authenticated-encryption modes. In: Joux, A. (ed.) FSE 2011. LNCS, vol. 6733, pp. 306–327. Springer, Heidelberg (Feb 2011); Cited on page 7.
[38] Leurent, G.: AEZ BBB. Rump session talk at Eurocrypt (2015); Cited on page 44.
[39] Luby, M., Rackoff, C.: How to construct pseudo-random permutations from pseudo-random functions (abstract). In: Williams, H.C. (ed.) CRYPTO'85. LNCS, vol. 218, p. 447. Springer, Heidelberg (Aug 1986); Cited on page 24.
[40] Minematsu, K.: Parallelizable rate-1 authenticated encryption from pseudorandom functions. In: Nguyen and Oswald [43], pp. 275–292; Cited on page 7.
[41] Namprempre, C.: Secure channels based on authenticated encryption schemes: A simple characterization. In: Zheng, Y. (ed.) ASIACRYPT 2002. LNCS, vol. 2501, pp. 515–532. Springer, Heidelberg (Dec 2002); Cited on page 33.
[42] Namprempre, C., Rogaway, P., Shrimpton, T.: Reconsidering generic composition. In: Nguyen and Oswald [43], pp. 257–274; Cited on pages 4, 8, 32, 33, 34, and 36.
[43] Nguyen, P.Q., Oswald, E. (eds.): EUROCRYPT 2014, LNCS, vol. 8441. Springer, Heidelberg (May 2014) ; Cited on page 29.
[44] NIST: FIPS 81: DES Modes of Operation. Issued December 2, 63 (1980); Cited on page 31.
[45] Okamoto, T. (ed.): ASIACRYPT 2000, LNCS, vol. 1976. Springer, Heidelberg (Dec 2000); Cited on page 28.
[46] Paterson, K.G., Ristenpart, T., Shrimpton, T.: Tag size does matter: Attacks and proofs for the TLS record protocol. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 372–389. Springer, Heidelberg (Dec 2011); Cited on page 34.
[47] Rogaway, P.: Authenticated-encryption with associated-data. In: Atluri [5], pp. 98–107; Cited on pages 24, 32, and 36.
[48] Rogaway, P.: Nonce-based symmetric encryption. In: Roy and Meier [52], pp. 348–359; Cited on pages 7, 9, 14, 24, 32, 35, and 36.
[49] Rogaway, P., Bellare, M., Black, J., Krovetz, T.: OCB: A block-cipher mode of operation for efficient authenticated encryption. In: ACM CCS 01. pp. 196–205. ACM Press (Nov 2001); Cited on pages 32 and 36.
[50] Rogaway, P., Shrimpton, T.: Cryptographic hash-function basics: Definitions, implications, and separations for preimage resistance, second-preimage resistance, and collision resistance. In: Roy and Meier [52], pp. 371–388; Cited on page 12.
[51] Rogaway, P., Shrimpton, T.: A provable-security treatment of the key-wrap problem. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 373–390. Springer, Heidelberg (May / Jun 2006); Cited on pages 8, 15, 19, 24, 32, 34, and 36.
[52] Roy, B.K., Meier, W. (eds.): FSE 2004, LNCS, vol. 3017. Springer, Heidelberg (Feb 2004); Cited on page 29.
[53] Sleevi, R., Watson, M.: Web Cryptography API. W3C Candidate Recommendation (2014), http://www.w3.org/TR/WebCryptoAPI/; Cited on page 5.
[54] Sun, Q., Simon, D.R., Wang, Y.M., Russell, W., Padmanabhan, V.N., Qiu, L.: Statistical identification of encrypted web browsing traffic. In: 2002 IEEE Symposium on Security and Privacy. pp. 19–30. IEEE Computer Society Press (May 2002); Cited on page 34.
[55] Tezcan, C., Vaudenay, S.: On hiding a plaintext length by preencryption. In: Lopez, J., Tsudik, G. (eds.) ACNS 11. LNCS, vol. 6715, pp. 345–358. Springer, Heidelberg (Jun 2011); Cited on page 34.
[56] Unger, N., Dechand, S., Bonneau, J., Fahl, S., Perl, H., Goldberg, I., Smith, M.: SoK: Secure messaging. In: 2015 IEEE Symposium on Security and Privacy. pp. 232–249. IEEE Computer Society Press (May 2015); Cited on page 7.
[57] Vaudenay, S.: Security flaws induced by CBC padding - applications to SSL, IPSEC, WTLS... In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 534–546. Springer, Heidelberg (Apr / May 2002); Cited on page 19.

A Historical Context

What constitutes authenticated encryption (and how to achieve it) has been widely studied. This section contains a historical overview, providing the context to interpret our results in terms of traditional notions of symmetric encryption. For consistency with the literature, these notions are presented with their original names, with $\mathcal{E}_k$ denoting an encryption oracle and $\mathcal{D}_k$ a decryption oracle. We use the subscript to denote secret material (the key), superscript to denote public material if there is any, and parenthesis to denote content to be encrypted or decrypted.

Probabilistic encryption. Initial formalisations of symmetric encryption relied strongly on the seminal work of Goldwasser and Micali on public key encryption [27]. Symmetric encryption is modelled as a probabilistic process that depends on a security parameter (typically provided indirectly by means of the private key). The standard *confidentiality* notion (cf. [32, Def. 3.30]) is indistinguishability under chosen plaintext attacks. An adversary with access to an honest encryption oracle throughout, should output (or find) two messages of equal length and, after receiving a random encryption of one of these, guess which of the two messages was encrypted by the challenge oracle. The scheme is deemed secure if all polynomial-time adversaries have a negligible success probability (all in terms of the security parameter). Randomness is required to prevent the encryption of equal inputs resulting in equal ciphertexts, which would clearly leak information about the inputs. This randomness in turn leads to expansion of the ciphertext.

The definition of security under chosen ciphertext attacks is similar, except that the adversary is additionally provided with an honest decryption oracle $\mathcal{D}_k$. Historically, two variants were considered depending on when the adversary may call $\mathcal{D}_k$: CCA1 (or lunch-time attack) refers to a scenario where the adversary no longer has access to its decryption oracle once it receives the challenge ciphertext, whereas CCA2 refers to everlasting access to the decryption oracle (which will suppress its output when given the challenge ciphertext). Nowadays, chosen ciphertext attacks (CCA) default to CCA2 (and the '2' is typically dropped).

For simplicity, the adversary is traditionally restricted to making only one query to its challenge oracle (resulting in only a single challenge ciphertext). One can easily extend the security notion to one where the adversary has unlimited access to its challenge oracle. This extension leads to an equivalent definition from the perspective of asymptotic security: a hybrid argument can be used to show that security under one notion implies security under the other (using a non-tight reduction).

Concrete security. Asymptotic security leads to a theoretically solid framework, but it poses problems when applied directly to practice. Most primitives underlying symmetric cryptology, for instance AES or Rijndael, do not come in infinite families (cf. RSA moduli in the public key setting), but rather only exist for a few choices of block length and key length. Formally, an asymptotic statement about AES is nonsensical, moreover a security reduction given purely in asymptotic terms cannot be used to determine the effect of block and key sizes on the security of a deployed scheme. To harness the hardness of a particular problem instantiation, an alternative called *concrete security* [10] is used instead. Security is expressed as the advantage that a resource-constrained adversary may at most achieve against a specific scheme. An encryption scheme is formalised as a triple key generation, encryption, and decryption $(\mathcal{K}, \mathcal{E}, \mathcal{D})$ and each security experiment begins by generating a key k with $\mathcal{K}$, (which the adversary is not allowed to see). The advantage can be thought of as the probability an adversary can distinguish between two possible experiments, or worlds.

Confidentiality. To measure *confidentiality*, or how hard it is for an adversary to learn the message from the ciphertext, Bellare et al. [10] define two security notions based on chosen plaintext attacks: left-or-right (LOR–CPA) and real-or-random (ROR–CPA). For both, the adversary is only provided access to a challenge encryption oracle. For the LOR–CPA game, the adversary submits a sequence of pairs $(x_{1,i}, x_{2,i})$ and must decide whether the oracle has been returning $\mathcal{E}_k(x_{1,i})$ or $\mathcal{E}_k(x_{2,i})$. On the other

hand, in the ROR–CPA game, the adversary submits a series of messages x_i and must decide whether the oracle returns $\mathcal{E}_k(x_i)$ or $\mathcal{E}_k(r_i)$, where $r_i \leftarrow\!\!\$\ \{0,1\}^{|x_i|}$. The LOR–CPA and ROR–CPA games are equivalent to each other up to a small factor in the advantage, and non-tightly to (a concrete variant of) the security game (as above) with a single challenge ciphertext and access to a true encryption oracle. These notions are collectively referred to as IND–CPA and are at the base of all modern confidentiality definitions.[4]

Authenticity. Bellare and Rogaway [15], as well as Katz and Yung [33], introduced an authenticity notion for probabilistic schemes, later dubbed INT–CTXT by Bellare and Namprempre [14]. The integrity (or *authenticity*) of a message implies the non-malleability goals of Dolev et al. [24], and (informally) means that the recipient can be confident that every message they receive is as the sender intended. Slightly more formally, the integrity of the ciphertext (INT–CTXT) convinces the receiver that the ciphertext is valid. In the INT–CTXT game, an adversary with access to $\mathcal{E}_k$ has to come up with a ciphertext C such that $\mathcal{D}_k$ accepts C, yet C was never output by the adversary's own $\mathcal{E}_k$ oracle. This directly implies the integrity of the plaintext (INT–PTXT), which captures that the decrypted plaintext has not been tampered with, although the reverse implication does not hold. One subtlety with defining INT–PTXT is that the presence of a verification oracle can significantly aid an adversary, whereas for ciphertext integrity this presence does not help that much [11, App. B].

Combining the two. Bellare and Namprempre [14] observe that IND–CPA and INT–CTXT together imply IND–CCA, often written as IND–CPA + INT–CTXT $\Rightarrow$ IND–CCA. The reverse is not true (IND–CCA security is possible for schemes that are not INT–CTXT secure), prompting Bellare and Namprempre to define authenticated encryption as the combination IND–CPA + INT–CTXT.

Although probabilistic encryption is conceptually neat, it does not (and never really did) capture practice, where encryption is generally instantiated with a deterministic algorithm that has access to either some state or to an external source of randomness. These two distinct scenarios are captured using stateful encryption and "Initialisation Vector" (IV) encryption, respectively. For the latter, security of the encryption ideally relies as little as possible on the quality of this external randomness, leading to nonce-based encryption.

Stateful encryption. In *stateful encryption*, key generation outputs an initial state in addition to the key. Encryption now takes as input a message, the key, and some state, and it outputs the ciphertext as well as a new state; decryption works similarly, also operating on a state (as well as taking in key and ciphertext and outputting a purported message). One key aspect of stateful encryption is that replay and reordering attacks should be prevented, which typically requires some form of synchronization between encryption and decryption states. The notion of synchronization also appears in the formalisation of stateful encryption due to Bellare et al. [13]. They generalise INT–CTXT to INT–sfCTXT by granting the adversary a win as soon as it makes a valid "out of order" decryption query, while for IND–sfCCA they allow *all* decryption queries, but suppress the output as long as the ciphertexts arrive in the exact same order as output by the (challenge) encryption oracle. They show that IND–CPA + INT–sfCTXT implies IND–sfCCA, effectively defining stateful authenticated encryption. Note that deterministic stateful schemes can still use randomness by including (the state of) a pseudorandom generator in the encryption and decryption states.

IV encryption. Classical encryption modes such as CBC, OFB, CFB, and CTR [44] all rely on an IV that is chosen anew each time a message is encrypted. While one could consider the IV as part of the ciphertext (and treat the scheme as a whole a probabilistic one), a formal model treating the IV as separate input to both encryption and decryption algorithm is preferable, e.g. to capture the very real

[4] IND–CPA is often sloppily referred to as semantic security, a notion which intuitively captures that from a ciphertext an adversary cannot learn anything about the underlying plaintext apart from possibly its length. Although asymptotically semantic security is equivalent to IND–CPA, formally it is quite a different notion.

possibility of the IV being transmitted out of band. A suitable syntax was pioneered by Rogaway and coauthors [47–49], who also introduce so-called *associated data*: data that should be authenticated but need not be encrypted, such as packet headers. From a security perspective, the decryption IV is always assumed to be under adversarial control, but there are three common options depending on how the IV is chosen for encryption queries. If it is chosen uniformly at random, *iv-based encryption* emerges; this can be considered a special case of probabilistic encryption. If the IV is unique but not necessarily random, it is referred to as a *nonce* (for "number used once"): an adversary can choose the encryption nonces in any way, as long as each nonce is used only once (this is called *nonce-respecting*). The resulting notion, *nonce-based encryption*, is strictly stronger than *iv-based encryption* (for large enough nonce spaces). Finally, if the adversary is allowed to repeat nonces, effectively rendering encryption entirely deterministic, the concept of *Misuse Resistant Authenticated Encryption* (MRAE) arises. This notion was explored by Rogaway and Shrimpton [51]. They defined nonce-based authenticated encryption using a single game dubbed IND$–CCA3 and proved IND$–CCA3 security is equivalent to security under both the IND$–CPA (see below) and INT–CTXT games (against nonce-respecting adversaries). By substituting uniqueness of nonces by uniqueness of the triplet IV–header–message they arrived at MRAE.

Random flavours. Capturing confidentiality, even under chosen plaintext attacks, can be done in essentially three different ways. The classical notion of indistinguishability IND (in its real-or-random incarnation) compares the encryption of the challenge message with the encryption of a random message of the same length.

An alternative definition of confidentiality is indistinguishability from random *ciphertexts*, or IND$ security. In the IND$ game, an adversary needs to distinguish between a challenge oracle that returns either the honest encryption or a ciphertexts of corresponding length, but drawn uniformly at random instead (i.e. a string of random bits of appropriate length). Thus security demonstrates that ciphertexts look random and so they cannot possibly leak information. For length-regular schemes (where $|\mathcal{E}_k(M)|$ depends on just $|M|$), IND$–CPA implies IND–CPA, but the converse does not hold.

The final notion of confidentiality does not look at individual ciphertexts, but instead considers the functionality as a whole. This leads to a security analogue for encryption of what pseudorandom permutations provide for blockciphers. A nonce-based encryption scheme syntactically is a family of injections, hence a secure encryption scheme should be indistinguishable from a (family of) truly random injections. Rogaway and Shrimpton [51, §8] termed this notion a pseudorandom injection (PRI). Along with the definition of an online cipher by Bellare et al. [8], this is an early example of a possible philosophical shift in AE definitions: instead of describing an idealised encryption (as with AE), they describe the "best possible" (a PRI).

Comparing paradigms. There are persuasive reasons for choosing any of the paradigms discussed above, both in relation to which syntax to use and what kind of security to aspire to. A brief discussion of the definitional merits of probabilistic, random-IV and nonce-based schemes is provided by Namprempre et al. [42]. We believe that, to a large extent, the choice depends on where to draw abstraction boundaries as we explain below.

Probabilistic versus random-IVs. Probabilistic schemes are self-contained and, once associated data is incorporated, have the most general syntax. Random IV schemes have a more restrictive syntax that better models reality, emphasizing that any randomness used for encryption is typically externally generated and not incorporated implicitly in the ciphertext, but rather sent explicitly (sometimes even out-of-band). Any random-IV scheme can be considered as a probabilistic scheme by prepending the (fixed-length) IV to the ciphertext. This transformation preserves security under the classical indistinguishability notions, converting any random-IV scheme into a similarly secure probabilistic scheme.

Conversely, a probabilistic scheme could be cast as an IV-based scheme, by abstracting out all use of randomness and declaring it the IV (rendering any randomness explicitly transmitted as part of the ciphertext superfluous). For most 'natural' secure probabilistic schemes this transformation works, resulting in a secure IV-based scheme. However, there are situations where the transformation is problematic,

either from an efficiency or security perspective (or both). When the randomness is implicit, casting the scheme to the IV model leads to unnecessary expansion. Consider for example the probabilistic Encode-then-Encipher scheme that given a secure $\pm\mathsf{prp}$ $\mathcal{E}$, encrypts by sampling some string I of known length v and outputting ciphertext $C \leftarrow \mathcal{E}_k(I||M||0^\tau)$ and decrypts by setting $I||M||T \leftarrow \mathcal{D}_k(C)$ and returning M iff $T = 0^\tau$. When turning this probabilistic scheme into a random-IV scheme, the ciphertext C cannot easily be contracted, thus transmission of the random string I creates additional overhead without clear benefit. In contrived examples, the randomness consumption of a probabilistic scheme might be message-dependent, or revealing the randomness could compromise security.

Random IVs versus nonces. Many schemes analysed as random-IV or probabilistic scheme require a truly randomised initial value for security. Once the IV can be predicted or manipulated by an adversary security could break down completely. CBC mode is a good example of a scheme relying on random IVs. Since in practice guaranteed true randomness is rare, reliance on true randomness can be problematic.

Nonce-based schemes, which share the same syntax as random-IV schemes, overcome this problem, as they only require unicity (at the encryption end) of initial vectors (now relabelled a nonce). Nonces are much easier to implement than uniform randomness, for instance using a counter. Moreover, a nonce-based scheme can generally be run with a randomised IV anyway, as for sufficiently large nonce spaces, drawing the IV uniformly at random will rarely lead to colliding nonces (the error term in the reduction being a standard birthday bound).

Nonces versus states. When implementing a nonce-based scheme using a counter to prevent reoccurrence of the nonce, the encryption algorithm necessarily needs to keep track of this counter, making it a *stateful* algorithm. However, only when decryption becomes stateful as well does it become appropriate to refer to Bellare et al.'s stateful security notions. The main difference between stateful security and nonce-based security is that the former explicitly rules out replay and reorder attacks (bringing it closer to the expectation of a secure channel [41]).

Nonce-based schemes trivially induce stateful schemes, where the state is initialized to zero during key generation and incremented each time a message is sent or received. This transformation preserves security (up to nonce wrap around): since the stateful decryption algorithm (re)computes the relevant nonces, replays and reordering is not possible.

While any probabilistic or IV scheme could be considered a (probabilistic) stateful scheme (by simply ignoring the state), it does not render a secure scheme; in particular replays and reordering will be trivial to achieve by an adversary.

Implications and separations. As we mentioned before, a lot of subtly different formalisations of indistinguishability have been proposed (real-or-random, left-or-right). For probabilistic encryption these notions have been shown equivalent, meaning security under one implies security under the other. However, they do not imply indistinguishability from random ciphertexts (IND$), which constitutes a separation. When implications and separations have been proven in one context (say probabilistic schemes), some care is required when interpreting them in another (say nonce-based schemes), as there is no a priori guarantee that *all* such results carry over (although most do quite straightforwardly), as discussed in Section 2.4. That said, several modern papers provide context-independent results (e.g. [3]), albeit not always in a fully formalised framework.

Modern approach. Recent trends in the literature support the view that the merits of a nonce-based treatment outweigh the limitations. For the remainder of this review, we will focus on this setting, where both encryption and decryption are deterministic and stateless functions of their inputs. This includes most efficient AE schemes, but rules out for instance stateful decryption [13] or fragmented decryption [18].

Tidiness. Namprempre et al. [42] introduced *tidiness* to the context of IV- (and thus nonce-) based encryption, defining a scheme to be tidy if decryption is a true inverse of encryption. That is, they extend the correctness requirement that $\mathcal{D}_k^{N,A}(\mathcal{E}_k^{N,A}(M)) = M$ by also requiring that $\mathcal{E}_k^{N,A}(\mathcal{D}_k^{N,A}(C)) = C$

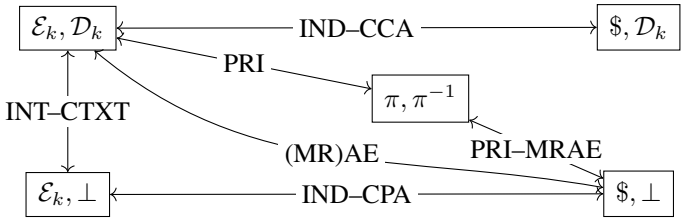

Fig. 5: Diagram of accepted security notions. Each notion is defined in terms of the adversarial advantage in distinguishing between appropriate pairs of oracles.

whenever $\mathcal{D}_k^{N,A}(C) \neq \perp$. Although rarely stated as an explicit design criteria, most existing schemes are tidy.

Thwarting traffic analysis. Traditional confidentiality notions (such as indistinguishability) take it for granted that any ciphertext will leak the length of its underlying plaintext (for some suitable and often implicit notion of length). In practice and in a larger context, plaintext lengths can be a valuable source of metadata that may (for instance) betray browsing behaviour [54]. To this end, two countermeasures, deployed in the real world, have been formalized.

Length-hiding encryption [46,55] seeks to disguise the length of messages by using variable amounts of redundancy to pad up to some predefined granularity. This granularity must be provided by entering the target ciphertext length to the encryption function (and allowing the encryption to fail if the ciphertext length is not supported given the other inputs). Given a ciphertext, the exact plaintext length can only be recovered with the decryption key. Length-hiding is a security goal that is seldom addressed by modern AE models, typically leaving this problem as one to be dealt with at the protocol level. Although length-hiding schemes need to support variable stretch (the difference between ciphertext and plaintext lengths), it should not be confused with variable stretch to enable user-defined authentication levels ([29], see also Section 3.3) as the latter requires transmitting the stretch with the ciphertext, which would render the scheme pointless in the length-hiding context.

Boundary-hiding encryption [18] seeks to disguise where one ciphertext ends and another one starts: ciphertexts arrive in a stream of random looking fragments. Given a sequence of concatenated ciphertexts, the number of involved plaintexts (let alone their lengths or content) can only be recovered with the decryption key. Dealing with arbitrarily fragmented ciphertexts leads to definitional challenges beyond just boundary-hiding [18, 26].

Authenticated Encryption as of 2014. We formally described the generally accepted security goals in Section 2.4, but briefly recall them here for completeness. Encryption is a deterministic algorithm, which takes both a nonce and (optional) associated data as input, and outputs a ciphertext. Decryption takes a nonce, associated data and ciphertext as input, and outputs either a message or a single error symbol. The tidiness requirement of Namprempre et al. [42] is generally met (and implicitly required for some results), and so likely to be an explicit requirement demanded of future schemes. The adversary is either nonce respecting (leading to AE security) or nonce abusing (leading to MRAE security), as defined by Rogaway and Shrimpton [51]. Security is often shown by separately bounding the IND–CPA and INT–CTXT advantages. These relations are represented in Fig. 5.

Explicit tag space. An alternative syntax for AE separates the ciphertext space into two components by splitting off an explicit tag space [2, 3]. We believe that the separation of tag and ciphertext as part of the syntax *only* makes sense if there is a clear definitional difference between the two, i.e. an algorithm that takes in one input but not the other or a security definition that assigns different meaning to the two spaces. For authenticated encryption this is not the case; for instance in the definitions of Andreeva et al. [3] the ciphertext and tag always appear side by side. Indeed, there are schemes for which no meaningful tag space can be defined (in particular schemes that add redundancy first and garble it

with the message, such as MAC-then-encrypt or encode-then-encipher [15]). For this reason, we do not include the concept of a tag space as part of our definitions. Of course when considering *constructions* it is perfectly reasonable to utilize "tagging" as part of the internal design, as used to great effect by the encrypt-then-MAC paradigm. Yet, the theoretical success of said paradigm does not justify cluttering the definitional framework.

B Sorting out the IND–CCA Nonce-sense

Defining a meaningful security notion is a balancing act. The strongest possible security notion contrasts a scheme with an alternative from which it is patently clear that the no-one could learn anything about the inputs, and demonstrates that no adversary can differentiate between the two. Yet, in order for the notion ever to be achieved, certain "unreasonable" adversarial strategies need to be disallowed. In general, the closer to ideal the reference world is and the more control we allow the adversary over the inputs the more adversaries, or more correctly the more adversarial behaviours, we must prohibit.

Traditionally, schemes were investigated in the probabilistic, LOR model. However, in this paper we have worked within a tidy, nonce-based IND$ based model (which we termed IND), and enforced some restrictions on the adversary not made in the traditional model. In this section, we will address these issues and confirm that security within our more expressive model does indeed lead to security in the sense expected by many practitioners. Firstly, we will briefly discuss how, in general terms, nonce-based IND security (following the IND$ definition of Rogaway [48]) relates to probabilistic LOR security (as defined by Bellare et al. [10]). Having done so, we will more thoroughly investigate the case of IND–CCA.

From IND to LOR. As we will state explicitly later in this section, a nonce-based scheme induces a probabilistic one that is also IND secure, albeit at the cost of a term that covers the probability of collisions among randomly sampled nonce values. Then, a probabilistic IND notion then implies LOR security, although the reduction loses a factor of 2 in the security bound. So, combining these two standard results, security in the nonce-based IND setting implies security in the LOR setting. However, as will become clear through the remainder of this section, one must be very careful about which sequences of queries are suppressed.

Prohibited Queries. In terms of experiment-based security definitions, preventing unreasonable adversarial behaviour means that certain sequences of queries must be prohibited (or, more generally, suppressed). One limitation we placed on the adversary (Section 2.3) was that they did not forward queries made to $\mathcal{D}_k$ to their Enc oracle. This means that having made a query $M \leftarrow \mathcal{D}_k(N, A, C)$, they do not then query $C' \leftarrow \mathrm{Enc}(N, A, M)$. After all, for tidy schemes $\mathcal{E}_k^{N,A}(\mathcal{D}_k^{N,A}(C)) = C$ so an adversary making such a query would trivially distinguish the Enc challenge oracle.

An important notion to which this restriction applies is IND–CCA. The IND–CCA advantage is equivalent to

$$\mathsf{Adv}^{\text{IND–CCA}}_{\mathcal{E},\Lambda}(\mathcal{A}) = \mathbb{P}\left[\mathcal{A}^{\mathcal{E}_k,\mathcal{D}_k} \to 1\right] - \mathbb{P}\left[\mathcal{A}^{\$,\mathcal{D}_k} \to 1\right].$$

We say a scheme is IND–CCA secure iff $\mathsf{Adv}^{\text{IND–CCA}}_{\mathcal{E},\Lambda}$ is sufficiently small for all (WLOG deterministic) adversaries $\mathcal{A}$ that do not forward queries between their oracles. So, after making a query to their first (Enc) oracle they do not make the corresponding query to their second ($\mathcal{D}_k$) oracle, *nor vice versa*.

This second restriction (that adversaries do not forward from $\mathcal{D}_k$ to Enc) is standard for IND–CCA definitions in the nonce-based literature, but is not made in the probabilistic paradigm. Now, any nonce-based scheme Π induces a random IV (and thus probabilistic) scheme $\mathcal{P}[\Pi]$ by uniformly sampling a nonce from N. However, as the nonce-based and probabilistic definitions are subtly different, it is unclear whether security of Π (as a nonce-based scheme) implies security of $\mathcal{P}[\Pi]$ (as a probabilistic scheme).

To resolve this, we will compare some variants of symmetric IND–CCA security relevant to nonce-based schemes. We will consider four security games, based on whether the (overall) scheme is nonce-based (n) or probabilistic (r, for random IV), and whether the (underlying) nonce-based scheme allows

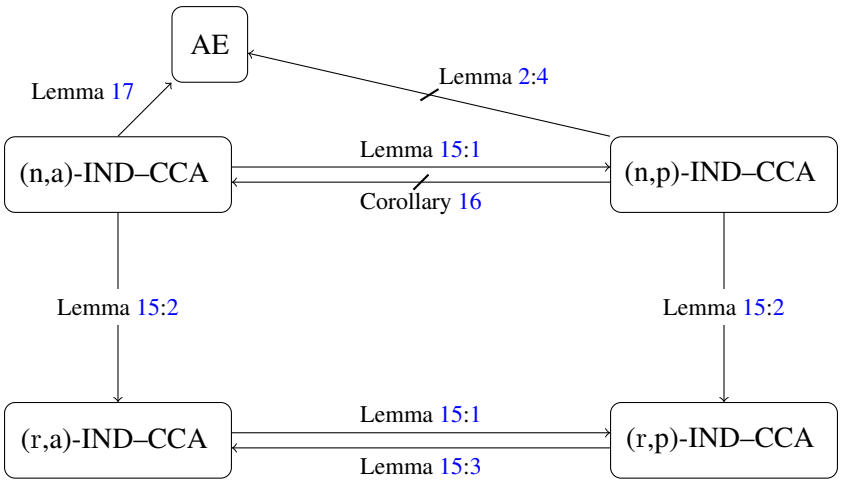

Fig. 6: Relation between different formalizations of IND–CCA encryption. The four security games are (x,y)-IND–CCA, where $\mathrm{x} \in \{\mathrm{n}, \mathrm{r}\}$ denotes whether the scheme is nonce-based or probabilistic (with a random IV), and $\mathrm{y} \in \{\mathrm{a}, \mathrm{p}\}$ denotes whether (underlying) nonce-based game allows or prohibits the adversary from forwarding queries from $\mathcal{D}_k$ to $\mathcal{E}_k$. The accepted nonce-based definition is (n,p) in the top-right, while the accepted probabilistic definition is (r,a) as shown bottom left. In particular, we observe that security in the nonce-based setting implies the security of the analogous probabilistic scheme.

(a) or prohibits (p) forwarding queries from $\mathcal{D}_k$ to Enc. We denote these as (x,y)-IND–CCA, with $\mathrm{x} \in \{\mathrm{n}, \mathrm{r}\}$ and $\mathrm{y} \in \{\mathrm{a}, \mathrm{p}\}$, and will investigate relations between them. Unless specified otherwise, we restrict ourselves to tidy, length-regular schemes with sufficiently large nonce-space.

We achieve a number of positive results (Lemma 15) most notably that (n,p)-IND–CCA security of Π implies the (r,a)-IND–CCA security of $\mathcal{P}[\Pi]$, then move on to investigate why (n,a)-IND–CCA is not the preferred definition for the nonce-based case (Lemma 17). These results, visualised in Figure 6, allow us to conclude that (n,p)-IND–CCA is indeed the logical choice of definition for IND–CCA in the nonce-based setting.

Finally, we note that many modern papers in the nonce-based setting do not consider IND–CCA directly, preferring to target AE, generally achieving this via the separated definition (e.g. [31, 42]).

Previous work. The definition of probabilistic LOR–CCA (previously CCA2) was introduced to the concrete, symmetric security setting by Bellare et al. [10, Definition 1]. Here, the adversary is prohibited from forwarding queries from Enc to $\mathcal{D}_k$, and these are the only prohibited queries. All other queries are allowed, although it is pointless to repeat queries to $\mathcal{D}_k$ (since it is deterministic). The same restrictions are used by Bellare and Namprempre [14, Definition 1] and Boldyreva et al. [19, Definition 5], and this is the accepted definition for games allowing multiple queries to the challenge oracle. For length-regular schemes, security under this left-or-right notion is implied by the corresponding indistinguishability from random bits notion, which is (r,a)-IND–CCA.

The nonce-based AEAD notion and games were refined over a number of papers [15, 47, 49] culminating at FSE 2004, where Rogaway defined the modern syntax and with it IND$–CCA [48, Section 6], the notion we term IND–CCA. The adversary (who does not make pointless queries) is forbidden from forwarding queries from Enc to $\mathcal{D}_k$, and *also vice versa*: they are prohibited from forwarding queries from $\mathcal{D}_k$ to Enc. In their DAE work, Rogaway and Shrimpton describe (but do not explicitly give) a definition for IND–CCA [51, Appendix B], by considering it as a weakening of nonce-based AE, but do not clarify which queries are to be prohibited (our interpretation suggests that no queries can be forwarded or repeated). The IND$–CCA notion of RUP allows the pointless repeat queries, but prohibits forwarding queries between the oracles [3, Definition 2]. Overall, when restricted to the effective adversary, the

accepted nonce-based definition is that which prohibits any repeating of queries or forwarding between the two oracles: (n,p)-IND–CCA.

In the public-key setting, Bellare et al. [12] investigated the subtle differences between various formalisations of IND–CCA (specifically, different flavours of IND–CCA2), and found they were not equivalent. Our investigation into IND–CCA notions in the symmetric setting is complementary; we look at different definitional choices as Bellare et al. They focused on a single challenge definition of indistinguishability and wondered how and when[5] to deal with an adversary wishing to query the decryption oracle on the challenge ciphertext. Their conclusion is that suppression on-the-fly provides to strongest notion, but whenever the adversary knows which queries will be suppressed, prohibiting these on-the-fly is equivalent.

We restrict our study to the multi-challenge setting where we prohibit queries on-the-fly (there exist alternative formalisations of symmetric CCA that allow just a single challenge, cf. [34, 36]) and concentrate on the differences between nonce-based versus probabilistic encryption and allowing versus prohibiting forwarding decryption outputs.

Relations amongst notions. Having established various definitions of security for both the nonce-based and probabilistic settings, we will now investigate how they relate to each other. Provided the nonce-space is large enough, nonce-based security implies probabilistic, moreover in the probabilistic setting allowing or prohibiting forwarding decryption queries to the challenge encryption oracle leads to equivalent notions. These fairly straightforward results are summarized in Lemma 15 (with further concrete advantage statements in the proof), leaving open only the status of (n,a) versus (n,p).

Lemma 15 (Implications between IND–CCA notions). *Let Π be a nonce-based* AE *scheme, with nonce-space* N. *Then for* $x \in \{\mathrm{n}, \mathrm{r}\}$ *and* $y \in \{\mathrm{a}, \mathrm{p}\}$,

1. (x,a)-IND–CCA $\implies$ (x,p)-IND–CCA
2. (n,y)-IND–CCA $\implies$ (r,y)-IND–CCA
3. (r,p)-IND–CCA $\implies$ (r,a)-IND–CCA

In particular, if Π is a secure nonce-based scheme, then the analogous probabilistic scheme $\mathcal{P}[\Pi]$ is secure, with

$$\mathsf{Adv}^{\text{(r,a)-IND–CCA}}_{\mathcal{P}[\Pi]}(q) \leq \mathsf{Adv}^{\text{(n,p)-IND–CCA}}_{\Pi}(q) + \frac{q_e(q+q_d)}{2 \cdot |\mathsf{N}|}$$

for any adversary making at most $q = q_e + q_d$ queries.

Proof (Sketch). Point 1 is trivial, since restricting the adversary cannot make him stronger.

Point 2 holds because, until $\mathcal{P}[\Pi]$ samples the same value for a second time, all queries made to Π have unique nonces, and thus security of Π implies security of $\mathcal{P}[\Pi]$. Explicitly,

$$\mathsf{Adv}^{\text{(r,y)-IND–CCA}}_{\mathcal{P}[\Pi]}(q) \leq \frac{q_e(q_e-1)}{2 \cdot |\mathsf{N}|} + \mathsf{Adv}^{\text{(n,y)-IND–CCA}}_{\Pi}(q).$$

Point 3 holds for similar reasons. Until the sampler picks a value that the adversary has already submitted as part of a decryption query, there is no way a query can be forwarded from decryption to encryption, giving the following explicit bound:

$$\mathsf{Adv}^{\text{(r,a)-IND–CCA}}_{\mathcal{P}[\Pi]}(q) \leq \frac{q_e \cdot q_d}{|\mathsf{N}|} + \mathsf{Adv}^{\text{(r,p)-IND–CCA}}_{\mathcal{P}[\Pi]}(q).$$

Combining these two bounds yields

$$\mathsf{Adv}^{\text{(r,a)-IND–CCA}}_{\mathcal{P}[\Pi]}(q) \leq \frac{q_e \cdot (2 \cdot q_d + q_e - 1)}{2 \cdot |\mathsf{N}|} + \mathsf{Adv}^{\text{(n,p)-IND–CCA}}_{\Pi}(q)$$

which, after simplification, yields the stated bound. □

[5] Prohibit versus suppress, resp. on-the-fly versus after the fact.

As is well-known, for length-regular schemes, (r,a)-IND–CCA security implies (r,a)-LOR–CCA security, which is the accepted definition of CCA security in the probabilistic LOR setting. Combined with the explicit bound of Lemma 15, this demonstrates that, if the nonce space is sufficiently large, IND-CCA security of a scheme in the nonce-based setting implies LOR-CCA security in the traditional probabilistic setting. For completeness, we state this as the following corollary.

Corollary 16. *A secure* IND–CCA *scheme induces a secure* LOR–CCA *scheme, provided its nonce-space is sufficiently large and can be sampled from uniformly.*

The case against (n,a)-IND–CCA. As shown by Corollary 16, security in the (n,p)-IND–CCA game is sufficient to yield security of the associated probabilistic scheme. However, this is at the additional cost of a collision event between encryption and decryption nonces (as well as the encryption-only collision required for the generic transformation from nonce-based to probabilistic). Perhaps the stronger (n,a)-IND–CCA provides a more direct and meaningful measure, that we should switch to instead. However, it turns out that (n, a)-IND–CCA is *equivalent* to AE itself (Lemma 17). Hence (n,a)-IND–CCA is arguably *too* strong a goal for IND–CCA security, because it leads to a collapse of the accepted hierarchy

Lemma 17. *Provided ϵ is not a valid ciphertext,* (n,a)-IND–CCA *is equivalent to* AE.

Proof. In a moment, we will prove that (n,a)-IND–CCA $\implies$ CTI–PAS. Previously, we showed that (n,a)-IND–CCA $\implies$ (n,p)-IND–CCA (Lemma 15:1) and (n,p)-IND–CCA + CTI–PAS $\iff$ AE (Theorem 6:2). Together, these three results demonstrate that (n,a)-IND–CCA $\implies$ AE as claimed. So, we now address the main step of this proof, demonstrating that (n,a)-IND–CCA security implies CTI–PAS security.

Let $\mathcal{A}$ be any adversary against CTI–PAS. We will construct $\mathcal{B}$ to be an adversary against (n,a)-IND–CCA, who runs $\mathcal{A}$ and simulates its environment. If $\mathcal{A}$ terminates then $\mathcal{B}$ also terminates, outputting 0. Every query $\mathcal{A}$ makes must be a Dec query (N, A, C), which $\mathcal{B}$ answers by forwarding it on to his own $\mathcal{D}_k$ oracle, which returns either $\perp$ or M. If at any point $\mathcal{D}_k$ returns some M, $\mathcal{B}$ proceeds to ask his challenge oracle to encrypt (N, A, M), receiving back some C'. $\mathcal{B}$ returns $C \stackrel{?}{=} C'$ to win the game.

If Enc is real then C' is always equal to C, because the scheme was tidy. If Enc is ideal then C' was uniformly sampled, and so will only equal C with probability $1/2^{|C|}$ (encryption is length-regular). Thus $\mathcal{B}$ wins the security game whenever $\mathcal{A}$ provides a valid ciphertext triple (N, A, C) and the ideal world does not "get lucky". Now, the probability of $\mathcal{A}$ outputting such a triple is precisely the advantage of $\mathcal{A}$ in the CTI–PAS game, since it is only by making such a query that the worlds can be distinguished. So, explicitly,

$$\begin{aligned}\mathsf{Adv}^{\text{(n,a)-IND–CCA}}_{\mathcal{E},\Lambda}(\mathcal{B}) &= \mathbb{P}\left[\mathcal{B}^{\mathcal{E}_k,\mathcal{D}_k} \to 1\right] - \mathbb{P}\left[\mathcal{B}^{\$,\mathcal{D}_k} \to 1\right] \\ &= \mathsf{Adv}^{\text{CTI–PAS}}_{\mathcal{E},\Lambda}(\mathcal{A}) \cdot 1 - \mathsf{Adv}^{\text{CTI–PAS}}_{\mathcal{E},\Lambda}(\mathcal{A}) \cdot 2^{-|C|} \\ &= \mathsf{Adv}^{\text{CTI–PAS}}_{\mathcal{E},\Lambda}(\mathcal{A}) \left(1 - 2^{-|C|}\right).\end{aligned}$$

Since ϵ is not a ciphertext, all ciphertexts contain at least one bit, implying $1 - 2^{-|C|} \geq 1/2$ or

$$\mathsf{Adv}^{\text{CTI–PAS}}_{\mathcal{E},\Lambda}(\mathcal{A}) \leq 2 \cdot \mathsf{Adv}^{\text{(n,a)-IND–CCA}}_{\mathcal{E},\Lambda}(\mathcal{B}).$$

Thus security in the (n,a)-IND–CCA game implies security in the CTI–PAS game, and so (n,a)-IND–CCA is equivalent to AE. □

In our proof we excluded the empty ciphertext, or phrased differently, assumed that all ciphertexts contain at least one bit. This is not a novel assumption [34], and, in the context of length-regular schemes, only prevents the particularly pointless scenario in which we attempt to "encrypt" the empty string with no expansion. To see that this exclusion is not spurious, let $\mathcal{E}_k$ be a secure AE scheme and define

$$\tilde{\mathcal{E}}^{N,A}_k(M) = \begin{cases} \varepsilon & M = \varepsilon \\ \mathcal{E}^{N,A}_k(M) & \text{Otherwise} \end{cases}$$

So, $\tilde{\mathcal{E}}_k$ acts identically to $\mathcal{E}_k$, except when called with the empty string ε, which it encrypts to ε.

On the one hand, $\tilde{\mathcal{E}}_k$ is still a secure IND–CCA scheme: since there is only one permutation from 0 bits to 0 bits (the constant function mapping ε to ε), $\tilde{\mathcal{E}}_k$ is no more distinguishable from a random length-regular function than $\mathcal{E}_k$. On the other hand, creating a forgery is trivial (ε is one) and thus winning the CTI–PAS game is trivial.

The equivalence of (n,a)-IND–CCA and AE implies that former is indeed stronger than (n,p)-IND–CCA, as stated in the corollary below. Instead of a separate proof, we simply refer to Lemma 2, where we showed that (n,p)-IND–CCA does not imply AE.

Corollary 18. (n,a)-IND–CCA *is strictly stronger than* (n,p)-IND–CCA.

The value of tidiness. One might wonder whether in the probabilistic setting a similar approach can be used to demonstrate that tidiness and (r,a)-IND–CCA imply AE as well. Without tidiness they do not, as shown by Bellare and Namprempre [14], so such a result would demonstrate that tidiness leads to an appreciable increase in security.

First, we must define what it means for a probabilistic scheme to be tidy. One reasonable definition would be to say a scheme is tidy iff whenever $\mathcal{D}_k^A(C) = M$, $C \in \text{Support}(\mathcal{E}_k^A(M))$. An immediate difference between this probabilistic definition and the nonce- or IV- based one is that in the probabilistic case tidiness does not imply decryption is a right-inverse of encryption. This means that we can no longer use the test $\mathcal{E}_k^A(\mathcal{D}_k^A(C)) \stackrel{?}{=} C$ to detect whether $\mathcal{D}_k$ or $\mathcal{E}_k$ has been replaced with their idealised version.

In the nonce case, our proof was split into two stages: first we showed IND-CCA implies CTI–PAS, then a composition result that IND–CCA + CTI–PAS $\implies$ AE. In the probabilistic setting, the composition still holds (assuming the appropriate restriction of queries), but the first implication does not hold. Our reduction was dependant on the validity-test described above, and without it one can construct a counterexample through the probabilistic analogue of Lemma 2:4.

Conclusion. Overall, we confirmed Rogaway's choice to forbid problematic queries. In doing so, nonce-based IND–CCA emerges as an interesting goal in its own right, preserving the separation that IND–CCA does not imply Authenticated Encryption, already well known from the probabilistic LOR–CCA setting [14], rather than causing a collapse in the hierarchy of security notions. We emphasise that IND–CCA defined thus provides indistinguishability (evidently) and thus confidentiality, yet no integrity (at least not guaranteed). From a cryptographic designer's point of view, this separation also means that, for adversaries that make neither prohibited nor pointless queries, the requirements on $\mathcal{D}_k$ and Dec are the same, neatly mirroring the similarities between $\mathcal{E}_k$ and Enc.

One may of course feel this whole discussion is moot, since most nonce-based schemes tend to avoid such controversy by directly targeting AE, either through a combined notion (in which case the separation between Enc and Dec is already given) or using the classical decomposition into IND–CPA and CTI–CPA (in which case it is not relevant). After all, most schemes known to achieve (nonce-based) IND–CCA security do so as a side effect of achieving AE anyway. In these cases, protection from the forgery-based attack above is given by an explicit integrity component, rather than as a possible implication of the confidentiality definition used.

C RUP's IND–CCA notions

As discussed in Section 3.2, RUP describes two notions of IND–CCA security. In each the adversary is provided with access to a challenge Enc oracle and the honest decryption/leakage function D_k, but neither provides access to V_k. The IND–CCA' game only prohibits the adversary from forwarding queries from $\mathcal{E}_k$ to $\mathcal{D}_k$, while IND–CCA also prohibits the adversary from forwarding queries from D_k to E_k. Lemma 8 asserted that RUP's IND–CCA' implies AE, while Lemma 9 said that but that RUP's IND–CCA does not even imply classical LOR–CCA, claims we now justify.

Alg. 0: Interfaces $\mathcal{B}$ provides $\mathcal{A}$

function ENC(N, A, M)
  return enc(N, A, M)
function DEC(N, A, C)
  $M \leftarrow \mathrm{D}_k(N, A, C)$
  $C' \leftarrow \mathrm{enc}(N, A, M)$
  if $C = C'$ **then**
    return M
  return $\perp$

Alg. 1: The case enc=E_k

function ENC(N, A, M)
  return $\mathrm{E}_k(N, A, M)$
function DEC(N, A, C)
  $M \leftarrow \mathrm{D}_k(N, A, C)$
  $C' \leftarrow \mathrm{E}_k(N, A, M)$
  if $C = C'$ **then**
    return M
  return $\perp$

Alg. 2: The case enc=\$

function ENC(N, A, M)
  return $\$(N, A, M)$
function DEC(N, A, C)
  $M \leftarrow \mathrm{D}_k(N, A, C)$
  $C' \leftarrow \$(N, A, M)$
  if $C = C'$ **then**
    bad $\leftarrow$ true
    return M
  return $\perp$

Fig. 7: Functionality used in Lemma 8 to simulate the oracles of the AE game when given access to either $\mathrm{E}_k, \mathrm{D}_k$ or $\$, \mathrm{D}_k$. The first algorithm lists the interfaces presented by $\mathcal{B}$ during it's simulation to $\mathcal{A}$. The second and third explicitly instantiate this with the two possible functions that $\mathcal{B}$s enc oracle might be representing.

Proof (Of Lemma 8). Let $\mathcal{A}$ be any AE adversary that does not make prohibited or pointless queries. We will construct $\mathcal{B}$ to be an IND–CCA' adversary running within similar resources to $\mathcal{A}$. The game will provide $\mathcal{B}$ with access to two oracles. One will be honest access to the D_k oracle, while the other, 'enc', will provide access to either E_k or \$. $\mathcal{B}$ will provide oracles (ENC,DEC) to $\mathcal{A}$, operating them as described in Algorithm 0. When $\mathcal{A}$ has terminated, $\mathcal{B}$ will forward the bit as his own answer.

Since $\mathcal{A}$ is an AE adversary, he does not repeat any queries or forward any queries between his oracles, and thus $\mathcal{B}$ can answer all queries without breaking his own limitations. We do require $\mathcal{B}$ be able to make enc queries corresponding to his own previous D_k queries, but this is allowed in RUP's IND–CCA' game. So, let us move on to calculating the probability of $\mathcal{B}$ winning the game.

In the case enc=E_k, we have that ENC=E_k, which by definition is equal to the encryption function $\mathcal{E}_k$ of the leak-free encryption scheme. Similarly, by tidiness we also have that DEC correctly implements the combined decrypt-verify function $\mathcal{D}_k$ of the encryption scheme. Therefore,

$$\mathbb{P}\left[\mathcal{B}^{\mathrm{E}_k,\mathrm{D}_k} \to 1\right] = \mathbb{P}\left[\mathcal{A}^{\mathcal{E}_k,\mathcal{D}_k} \to 1\right]$$

In the case enc=\$, we have that ENC=\$. More interestingly, unless bad is set, DEC=$\perp$. So, writing DEC[\$] for the oracle DEC instantiated around \$ and using β as shorthand for $\mathbb{P}[\mathsf{bad}]$,

$$\mathbb{P}\left[\mathcal{B}^{\$,\mathrm{D}_k} \to 1\right] = (1-\beta)\cdot\mathbb{P}\left[\mathcal{A}^{\$,\perp} \to 1\right] + \beta\cdot\mathbb{P}\left[\mathcal{A}^{\$,\mathrm{DEC}[\$]} \to 1|\mathsf{bad}\right].$$

While it is unclear quite how one might calculate $\mathbb{P}\left[\mathcal{A}^{\$,\mathrm{DEC}[\$]} \to 1|\mathsf{bad}\right]$, this will not be necessary. The IND–CCA' advantage of $\mathcal{B}$ is the difference between these two terms, and so,

$$\begin{aligned}
\mathsf{Adv}^{\text{IND–CCA'}}_{(\mathrm{E},\mathrm{D})}(\mathcal{B}) &= |\ \mathbb{P}\left[\mathcal{A}^{\mathcal{E}_k,\mathcal{D}_k} \to 1\right] - \mathbb{P}\left[\mathcal{A}^{\$,\perp} \to 1\right] - \beta\cdot(\ \mathbb{P}\left[\mathcal{A}^{\$,\mathrm{DEC}(\$)} \to 1|\mathsf{bad}\right] - \mathbb{P}\left[\mathcal{A}^{\$,\perp} \to 1\right]\ )\ | \\
&= |\quad \mathsf{Adv}^{\mathrm{AE}}_{\mathcal{E}}(\mathcal{A}) \quad - \beta\cdot(\ \mathbb{P}\left[\mathcal{A}^{\$,\mathrm{DEC}(\$)} \to 1|\mathsf{bad}\right] - \mathbb{P}\left[\mathcal{A}^{\$,\perp} \to 1\right]\ )\ | \\
&\geq \quad |\mathsf{Adv}^{\mathrm{AE}}_{\mathcal{E}}(\mathcal{A})| \quad - \beta\cdot|\ \mathbb{P}\left[\mathcal{A}^{\$,\mathrm{DEC}(\$)} \to 1|\mathsf{bad}\right] - \mathbb{P}\left[\mathcal{A}^{\$,\perp} \to 1\right]\ | \\
&\geq \quad \mathsf{Adv}^{\mathrm{AE}}_{\mathcal{E}}(\mathcal{A}) \quad - \beta.
\end{aligned}$$

For the penultimate inequality we have used the triangle inequality, and for the final that the difference between the probabilities must be at most 1.

Consider briefly the adversary $\mathcal{C}$ against E under IND–CCA' that simply asks for the encryption of messages yielding the shortest ciphertexts[6] and looks for a collision in the output. This never occurs

[6] One may wonder how $\mathcal{C}$ can discover what message length this corresponds to. For all natural length-regulation functions τ it is trivial to find the minimum value. In the more complex (and highly contrived) case that finding the minima of τ is hard, it suffices that $\mathcal{C}$ ask for the encryption of 0^i for $1 \leq i \leq 128$. Since $\tau(X) \geq |X|$ (or the scheme could not be correct), this suffices to find any possible ciphertext lengths below 128 bits. If there are no such ciphertexts, the probability β is sufficiently small anyway not to be a problem.

Alg. 3: Encryption oracle $\tilde{\mathrm{E}}_{k||l}$

function $\tilde{\mathrm{E}}_{k||l}(N, M)$
  if $M = \mathcal{F}_l(N)$ **then**
    $\mathsf{bad}_{\mathrm{PRF}} \leftarrow \mathtt{true}$
    return $k_N||0^{1+\tau+\gamma}$
  $C \leftarrow \mathrm{E}_k^N(00||M)$
  if $C = b||0^{1+\tau+\gamma}, b \in \{0,1\}$ **then**
    $\mathsf{bad}_{\mathrm{IND}} \leftarrow \mathtt{true}$
    $C \leftarrow \mathrm{E}_k^N(10||M)$
    if $C = \bar{b}||0^{1+\tau+\gamma}$ **then**
      $C \leftarrow \mathrm{E}_k^N(11||M)$
  return C

Alg. 4: Separated Decryption oracle $\tilde{\mathrm{D}}_{k||l}$

function $\tilde{\mathrm{D}}_{k||l}(N, C)$
  if $C = k_N||0^{1+\tau+\gamma}$ **then**
    return $\mathcal{F}_l(N)$
  if $C = \overline{k}_n||0^{1+\tau+\gamma}$ **then**
    return $\mathcal{F}_l(N)$
  $x||y||M \leftarrow \mathrm{D}_k^N(C)$
  return M

Alg. 5: Combined Decrypt&Verify Oracle $\tilde{\mathcal{D}}_{k||l}$

function $\tilde{\mathcal{D}}_{k||l}(N, C)$
  if $C = k_N||0^{1+\tau+\gamma}$ **then**
    return $\mathcal{F}_l(N)$
  if $C = \overline{k}_n||0^{1+\tau+\gamma}$ **then**
    return $\perp$
  if $\mathrm{V}_k^N(C) = \perp$ **then**
    return $\perp$
  $x||y||M \leftarrow \mathrm{D}_k^N(C)$
  if $x = 0, y = 0 \wedge M \neq \mathcal{F}_l(N)$ **then**
    return M
  if $x = 1$ **then**
    $\mathsf{bad}_{\mathrm{IND}} \leftarrow 1$
    $C_{00} \leftarrow \mathrm{E}_k^N(00||M)$
    $C_{10} \leftarrow \mathrm{E}_k^N(10||M)$
    if $C_{00} = b||0^{1+\tau+\gamma}, b \in \{0,1\}$ **then**
      if $y = 0 \wedge C_{10} \neq \bar{b}||0^{1+\tau+\gamma}$ **then**
        return M
      if $y = 1 \wedge C_{10} = \bar{b}||0^{1+\tau+\gamma}$ **then**
        return M
  return $\perp$

Fig. 8: The RUP scheme $\tilde{\mathrm{E}}, \tilde{\mathrm{D}}, \tilde{\mathrm{V}}$. We explicitly give $\tilde{\mathrm{E}}$ and $\tilde{\mathrm{D}}$, and provide the combined decrypt-and-verify oracle $\tilde{\mathcal{D}}$. The verification oracle $\tilde{\mathrm{V}}$ can be derived from $\tilde{\mathcal{D}}$ by running $\tilde{\mathcal{D}}$ and returning $\perp$ if $\tilde{\mathcal{D}}$ did, and $\top$ else. The variables b, x, y are all one bit long.

in the $\mathcal{E}_k$ case, but may occur in the \$ case. Thus the probability of $\mathcal{C}$ distinguishing the schemes is precisely that of bad occurring, and so bounds β. So, rearranging the inequality above and substituting in the bound for β, we have that

$$\mathsf{Adv}^{\mathrm{AE}}_{\mathcal{E}}(\mathcal{A}) \leq \mathsf{Adv}^{\text{IND–CCA'}}_{(\mathrm{E}_k,\mathrm{D}_k)}(\mathcal{B}) + \beta \leq \mathsf{Adv}^{\text{IND–CCA'}}_{(\mathrm{E}_k,\mathrm{D}_k)}(\mathcal{B}) + \mathsf{Adv}^{\text{IND–CCA'}}_{(\mathrm{E}_k,\mathrm{D}_k)}(\mathcal{C}) \leq 2 \cdot \mathsf{Adv}^{\text{IND–CCA'}}_{(\mathrm{E}_k,\mathrm{D}_k)}$$

Thus IND–CCA' security implies AE security of the leak-free scheme. □

Lemma 9 asserted that RUPs IND–CCA notion did not imply traditional IND–CCA. To show this, we will describe a scheme that is secure under RUP's IND–CCA notion (henceforth RUP-IND–CCA), but insecure under classical IND–CCA. We do so under the assumption that there exists a secure RUP-IND–CCA scheme, and a secure PRF (which can be bootstrapped from the RUP-IND–CCA scheme). We begin with an informal description of the construction, before providing a more thorough evaluation.

Given an encryption scheme E and a pseudo-random function $\mathcal{F}$, we will construct an encryption scheme $\tilde{\mathrm{E}}$ taking in two keys k, l that acts very similarly to E, except that $\tilde{\mathrm{E}}^i_{k||l}(\mathcal{F}_l(i)) = k_i||0^{1+\tau+\gamma}$. In the classical IND–CCA game, requesting $\tilde{\mathcal{D}}^i_{k||l}(1||0^{1+\tau+\gamma})$ will return the i^{th} key bit, meaning after κ queries the adversary has the whole key and can trivially distinguish the scheme. However, by defining $\Lambda_k(\overline{k_i}||0^{1+\tau+\gamma}) = \mathcal{F}_l(i)$, we can ensure that in the RUP-IND–CCA setting $\tilde{\mathrm{D}}^i_{k||l}(b||0^{1+\tau+\gamma}) = \mathcal{F}_l(i)$ no matter whether the adversary guesses $b = 0$ or $b = 1$. So, this behaviour occurs independent of the key bit, and so in the RUP case the adversary cannot exploit the weakness, meaning security is inherited from E.

Stating this result more formally requires additional measures to ensure correctness of the scheme, and the appropriate security reductions to prove this really is the only weakness that has been added. These mean the actual counter-example is significantly more cumbersome. Its definition and analysis are given in the following proof.

Alg. 6: Oracles $\mathcal{O}_E$ and $\boxed{\mathcal{O}_\$}$

function $\mathcal{O}_E$, $\mathcal{O}_\$(N, M)$
- $C \leftarrow \mathrm{E}_k^N(00||M)$
- $\boxed{C \leftarrow \$^N(00||M)}$
- **if** $M = \mathcal{F}_l(N)$ **then**
  - $\mathsf{bad}_{\mathrm{PRF}} \leftarrow \mathtt{true}$
- **if** $C = b||0^{1+\tau+\gamma}, b \in \{0,1\}$ **then**
  - $\mathsf{bad}_{\mathrm{IND}} \leftarrow \mathtt{true}$
- **return** C

Alg. 7: Oracle $\mathcal{O}_D$ and $\tilde{\mathrm{D}}_{k||l}$ are identical

function $\mathcal{O}_D(N, C)$
- **if** $C = b||0^{1+\tau+\gamma}, b \in \{0,1\}$ **then**
  - **return** $\mathcal{F}_l(N)$
- $b||c||M \leftarrow \mathrm{D}_k^N(C)$
- **return** M

Fig. 9: Oracle $\mathcal{O}_E$ does not contain the boxed code, while Oracle $\mathcal{O}_\$$ does. The pair of oracles $(\mathcal{O}_E, \mathcal{O}_D)$ are identical-until-bad to $(\tilde{\mathrm{E}}_{k||l}, \tilde{\mathrm{D}}_{k||l})$. The pair $(\mathcal{O}_\$, \mathcal{O}_D)$ differ from $(\mathcal{O}_{\mathcal{E}_k}, \mathcal{O}_D)$ by at most the RUP-IND–CCA advantage of an adversary against (E, D).

Proof (Of Lemma 9). Let $(\mathrm{E}, \mathrm{D}, \mathrm{V})$ be a scheme secure under RUP's IND–CCA notion, with fixed stretch τ and $\mathsf{K} = \{0,1\}^\kappa$, where $|\mathsf{N}| > \kappa$. Let $\mathcal{F} : \mathsf{K} \times \{1, \ldots, \kappa\} \to \{0,1\}^\gamma$ be a secure PRF (this can be built from E). Define the RUP scheme $(\tilde{\mathrm{E}}, \tilde{\mathrm{D}}, \tilde{\mathrm{V}})$ as in Figure 8. For conciseness, we will be omitting associated data from our notation.

Let us consider first the traditional IND–CCA case, in which the adversary has access to an Enc oracle implementing either $\tilde{\mathcal{E}}_{k||l}$ or \$, and a decryption $\tilde{\mathcal{D}}_{k||l}$ oracle. In this case, the scheme is not secure because the adversary can learn the key through decryption queries. Explicitly, there exists an adversary $\mathcal{A}$ who makes the queries $\tilde{\mathcal{D}}^i_{k||l}(1||0^{\tau+1})$ for each $i = 1, \ldots, \kappa$. If the ciphertext was invalid (ie $\tilde{\mathcal{D}}_{k||l}$ returned $\perp$) then $\mathcal{A}$ can conclude $k_i = 0$, and if it was valid then $k_i = 1$. So, after κ queries $\mathcal{A}$ has learnt the whole key, and can trivially distinguish the scheme.

We move on to RUP-IND–CCA security. In this the adversary is given access to oracles $(\tilde{\mathrm{E}}_{k||l}, \tilde{\mathrm{D}}_{k||l})$, but not to $(\tilde{\mathrm{V}}_{k||l})$. Moreover, they are prohibited from forwarding queries between their two oracles. We will prove the scheme secure for all adversaries making at most q queries. First, with a very simply identical until bad argument, we will simplify the schemes to form $(\mathcal{O}_E, \mathcal{O}_D)$ given in Figure 9. Next, we bound the probability of the first bad event, $\mathsf{bad}_{\mathrm{PRF}}$ being set by $\mathcal{O}_E$, using the PRF security of $\mathcal{F}_l$. Then we bound the probability of $\mathsf{bad}_{\mathrm{IND}}$ being set by an adversary interacting with $(\mathcal{O}_E, \mathcal{O}_D)$. To do so, we appeal to the RUP-IND–CCA security of $(\mathrm{E}_k, \mathrm{D}_k)$ to swap them for their idealised versions, which also proves $(\mathcal{O}_E, \mathcal{O}_D)$ secure, completing the proof.

Consider the probability of an adversary interacting with $(\mathcal{O}_E, \mathcal{O}_D)$ setting $\mathsf{bad}_{\mathrm{PRF}}$. This can only occur on an encryption query, and requires the adversary to submits a nonce and message such that $M = \mathcal{F}_l(N)$. The adversary does not repeat queries nor forward queries from his $\tilde{\mathrm{D}}_{k||l}$ oracle to $\tilde{\mathrm{E}}_{k||l}$, so this must be the first time he has queried (N, M). In particular, this means he cannot have previously queried $M \leftarrow \mathcal{O}_D(N, b||0^{1+\tau+\gamma})$ for either $b \in \{0,1\}$, and so can learn nothing useful from his decryption oracle. Thus the most efficient adversary seeking to set $\mathsf{bad}_{\mathrm{PRF}}$ will only interact with the encryption oracle. Combining this with the fact $\mathcal{F}$ is keyed independently from the internal encryption routine E (they use keys l and k respectively), we can swap out $\mathcal{F}_l$ for its idealised version, a random function, at the cost of $\mathbf{Adv}^{PRF}_{\mathcal{F}}(q)$. So, the probability of $\mathsf{bad}_{\mathrm{PRF}}$ is clearly bounded by $q/2^\gamma$.

At the cost of the RUP-IND–CCA advantage against $(\mathrm{E}_k, \mathrm{D}_k)$, we now switch E_k for its idealised version, \$, to form a pair of oracles $(\mathcal{O}_\$, \mathcal{O}_D)$, where $\mathcal{O}_\$$ is simply $\mathcal{O}_E$ after this switch. Now, $\mathbb{P}\left[\mathcal{A}^{\mathcal{O}_\$} \text{sets } \mathsf{bad}_{\mathrm{IND}}\right] \leq q/2^{1+\tau+\gamma}$, since it is the probability of a random function hitting a single value. Finally then, we observe that this pair is in fact the ideal world with which we are comparing $(\tilde{\mathrm{E}}_{k||l}, \tilde{\mathrm{D}}_{k||l})$.

So, it remains to collect the separate terms to form the final bound:

$$\mathsf{Adv}^{\text{IND–CCA}}_{(\tilde{\mathrm{E}}_{k||l}, \tilde{\mathrm{D}}_{k||l})}(q) \leq \mathsf{Adv}^{\mathrm{PRF}}_{\mathcal{F}}(q) + \frac{q}{2^\gamma} + \mathsf{Adv}^{\text{IND–CCA}}_{(\mathrm{E},\mathrm{D})}(q) + \frac{q}{2^{1+\gamma+\tau}}.$$

As $\mathcal{F}$ was a secure PRF and (E, D) secure under the RUP-IND–CCA notion, both these terms are small. Taking sufficiently large γ, this proves the overall RUP-IND–CCA security of $(\tilde{\mathrm{E}}, \tilde{\mathrm{D}})$. □

D Direct Equivalence of RUP and RAE[τ]

It is also possible to directly compare the RUP and RAE[τ] notions. They were clearly designed to quantify robustness for two very different classes of design: whilst the RUP paper focusses on its application to online schemes, RAE[τ] expands security notions to cover variable stretch schemes. The two notions are both applicable to traditional fixed stretch offline schemes, and it is in this context that we can compare them. This then becomes a case of bounding the difference between the ideal worlds, and we find them to be equivalent.

Translating RAE[τ] into the RUP syntax. We can re-set the RAE[τ] security game into the three-oracle form favoured by RUP. Let $(\mathcal{E}_k, \mathcal{D}_k)$ be the candidate two-oracle scheme and S the appropriate simulator. Then define the separation of the scheme to be $(\mathrm{E}_k, \mathrm{D}_k, \mathrm{V}_k)$ and the separation of the ideal world to be $(\pi, \mathsf{S}_\pi, \mathrm{V}_\pi)$ as shown in Figure 10. This separation is intuitive, being little more than notational, and similar to the translation used in Section 3.3. It allows us to state the RAE[τ] advantage in RUP syntax as

$$\mathsf{Adv}^{\mathrm{RAE}[\tau]}_{\mathcal{E},\Lambda} := \Delta^{\mathrm{E}_k,\mathrm{D}_k,\mathrm{V}_k}_{\pi\ ,\mathsf{S}_\pi,\mathrm{V}_\pi}.$$

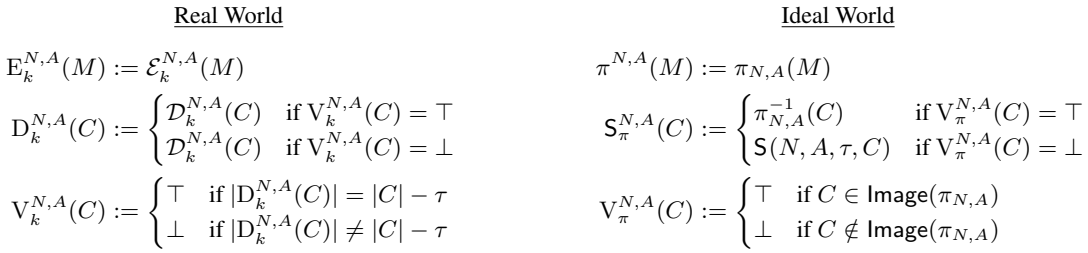

Fig. 10: Translating an RAE[τ] scheme and the RAE[τ] ideal world to the separated AE setting.

Comparing the two notions. In Lemma 13, we showed that if there exists any good RAE[τ] simulator then Λ_k is one. So, the key security objectives of the papers are:

$$\mathsf{Adv}^{\mathrm{RUPAE}}_{\mathcal{E},\Lambda} := \Delta^{\mathrm{E}_k,\mathrm{D}_k,\mathrm{V}_k}_{\$\ ,\mathrm{D}_l,\ \bot} \approx \mathsf{Adv}^{\mathrm{RUP}}_{\mathcal{E},\Lambda} + \mathsf{Adv}^{\mathrm{DI}}_{\mathcal{E},\Lambda} + \mathsf{Adv}^{\mathrm{CPA}}_{\mathcal{E},\Lambda}$$
$$\mathsf{Adv}^{\mathrm{RAE}[\tau]}_{\Pi,\Lambda} := \Delta^{\mathrm{E}_k,\mathrm{D}_k,\mathrm{V}_k}_{\pi\ ,\mathsf{S}_\pi,\mathrm{V}_\pi}$$

where $\$$ is a random function, π a random injection, and $\mathrm{V}_k, \mathrm{V}_\pi$ honest verifiers with respect to the key k or injection π. whilst S_π is an oracle who forwards queries to either its simulator (i.e. Λ) or to π, as defined in Figure 10. When written in this form, it is not surprising that they are in fact so similar.

Theorem 19. *RUPAE and RAE[τ] security are equivalent. In particular,*

$$\left|\mathsf{Adv}^{\mathrm{RUPAE}}_{\mathcal{E},\Lambda} - \mathsf{Adv}^{\mathrm{RAE}[\tau]}_{\Pi,\Lambda}\right| \leq \Delta^{\emptyset,\emptyset,\mathrm{V}_\pi}_{\emptyset,\emptyset,\ \bot} + \Delta^{\pi,\emptyset,\emptyset}_{\$,\emptyset,\emptyset} \leq 1.5\frac{q}{2^\tau}$$

for any adversaries making at most q queries, where the stretch τ is fixed.

Proof. By the triangle inequality,

$$\left|\mathsf{Adv}^{\mathrm{RUPAE}}_{\mathcal{E},\Lambda} - \mathsf{Adv}^{\mathrm{RAE}[\tau]}_{\Pi,\Lambda}\right| = \Delta^{\pi,\mathsf{S}_\pi,\mathrm{V}_\pi}_{\$,\Lambda_l,\ \bot} \leq \Delta^{\pi,\mathsf{S}_\pi,\mathrm{V}_\pi}_{\pi,\Lambda_l,\ \bot} + \Delta^{\pi,\Lambda_l,\bot}_{\$,\Lambda_l,\bot}$$

The first term here is bounded above by $\Delta^{\pi,\Lambda_l,\mathrm{V}_\pi}_{\pi,\Lambda_l,\perp}$. This is because until the adversary can win this forging game the output of both simulators is the same, since he has not found a query $c \in \mathsf{Image}(\pi)$. So, the middle oracle does not actually assist him, and because π is random, domain separation means the π oracle does not help either. Thus this reduces to sampling with no oracles, and hoping to find some $c \in \mathsf{Image}(\pi)$. Within q queries, this is bounded above by $q/2^\tau$, and corresponds to the classical "tag guessing" strategy.

The second term is simply a variable length random injection to random function switch, and bounded by the PRI–MRAE difference. Alternatively, we can bound it directly. Since a random function and random injection are indistinguishable until an output collision occurs, we can sum over all possible message lengths m and the number of queries q_m made of that length. Since there are at most 2^m messages of length m, we have $q_m \leq 2^m$ and so

$$\Delta^{\pi}_{\$}(q) \leq \sum_m \frac{q_m(q_m-1)}{2^{m+\tau+1}} = \sum_m \frac{q_m}{2^{\tau+1}} \cdot \frac{q_m - 1}{2^m} \leq \frac{q}{2^{\tau+1}}$$

Combining these,

$$\begin{aligned}\left|\mathsf{Adv}^{\mathrm{RUPAE}}_{\mathcal{E},\Lambda} - \mathsf{Adv}^{\mathrm{RAE}[\tau]}_{\Pi,\Lambda}\right| &\leq \Delta^{\pi,\mathsf{S}_\pi,\mathrm{V}_\pi}_{\pi,\,\mathsf{S}\,,\perp} + \Delta^{\pi,\mathsf{S},\perp}_{\$,\mathsf{S},\perp} \\ &\leq \Delta^{\emptyset,\emptyset,\mathrm{V}_\pi}_{\emptyset,\emptyset,\perp} + \Delta^{\pi,\emptyset,\emptyset}_{\$,\emptyset,\emptyset} \\ &\leq 1.5\tfrac{q}{2^\tau}\end{aligned}$$

which is the bound as claimed. □

E Early–abort AEZ is not RAE Secure

The bulk of this paper has discussed the definitional merits of different ways to describe security in the presence of leakage from decryption. This appendix is somewhat perpendicular to this objective, but included to illustrate just how careful one must be when modelling unwanted decryption leakage.

AEZ [29] was the first scheme proven to provide RAE security. This was based around a hybrid argument: AEZ is a secure ±PRP, and encode-then-encipher around an ±PRP provides RAE security. However, the implementation section [29, §10] observes that one can reject invalid ciphertexts early, saving the cost of running the second half of the decryption routine:

> Another benefit of AEZ's two passes is that the second pass is not needed to discover that a ciphertext is inauthentic, leading to message rejection costing as little as 0.28 cpb on Haswell. On long messages, approximately 2/5 of AES4 calls are performed during the first pass, which aligns perfectly with the peak times we've observed for encryption and fast-rejection.

In the single error context (i.e. without decryption leakage), it is clear this optimised decryption algorithm is functionally equivalent to the full algorithm. However, this is not the case when decryption leakage is allowed. The RAE security claims against which AEZ is tested provide the adversary access to the final contents of the internal buffer upon an invalid decryption query. If the full AEZ has been run, the contents of this buffer are randomised and do not leak useful data to the adversary. However, this is not the case when the scheme is aborted early.

When the scheme early-aborts, the intermediate buffer will contain leakage information. In particular, for the main bulk of the encryption, it will contain (Y_i, Y_i'), where Y_i' is the value corresponding to Y_i, but on the other wire.

Against the published version of AEZ referenced above, this observation can be combined with Leurent's attack [38] to recover the master key recover if using the AEZ–prf. Later versions of AEZ have been modified to defend against Leurent's attack by strengthening the tweakable PRF, but we emphasise this attack is against the structure of early–abort AEZ itself and thus still applies to the current version AEZ-v4.1 [28].

The solution to this problem is a simple (yet important) one: the documentation of any potentially robust Authenticated Encryption scheme must be completely clear as to which claims apply to which

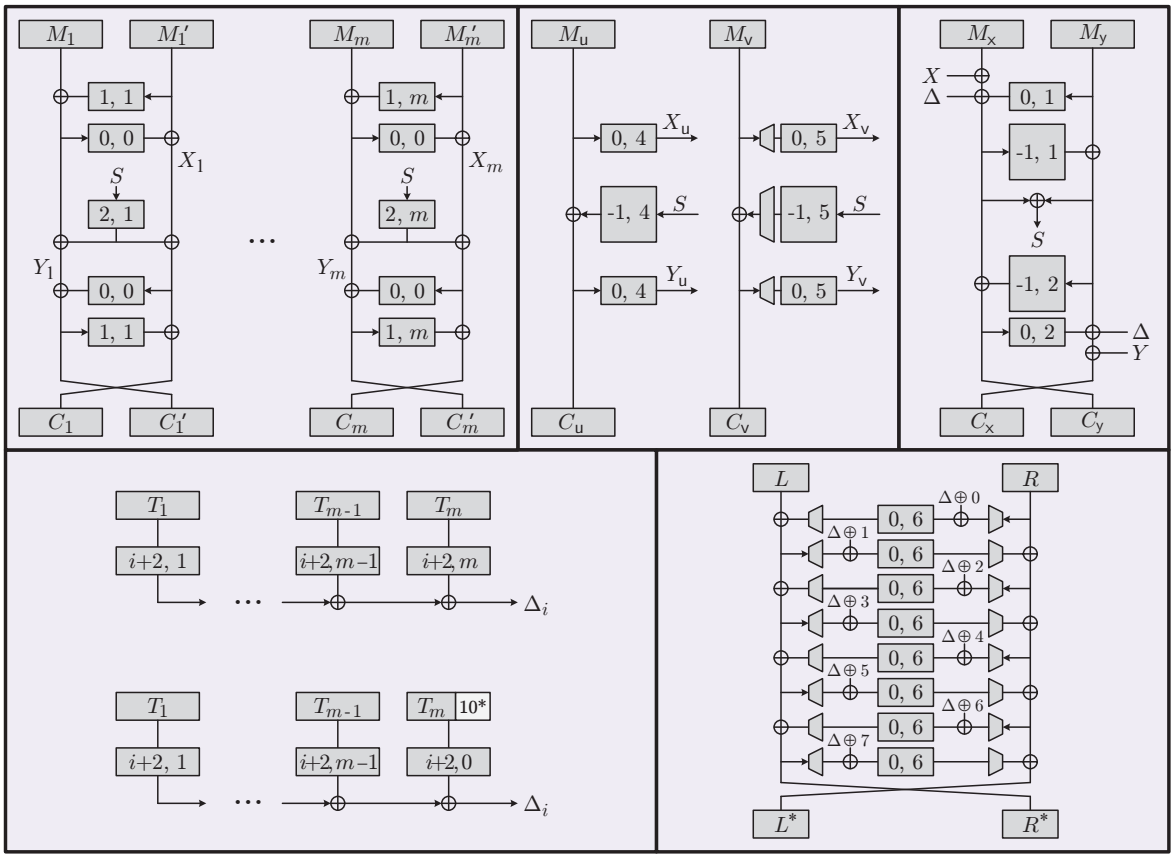

Fig. 11: Diagram of AEZ [29, Figure 7]. The specification observes that having calculated Δ and $Y = \oplus_i Y_i$, one can decrypt the final two blocks, and in doing so check the padding and thus validity of the ciphertext. Computation is then allowed to abort early, meaning the "main branches" will not be evaluated any further "up" the diagram than Y_i.

implementation. To put the corrected statement into our terminology, let $\mathrm{E}_k, \mathrm{D}_k$ be the encryption and decryption functions of AEZ, and Λ_k the leakage function that exposes the final internal state after full evaluation of the decryption function (the string $M_1 M_1' \cdots M_x M_y$). Then $(\mathrm{E}_k, \mathrm{D}_k, \Lambda_k)$ is a secure SAE or RAE scheme, while $(\mathrm{E}_k, \mathrm{D}_k, \emptyset)$ is a secure AE scheme with fast rejections of invalid ciphertexts.

Forgery against AEZ under RAE with early–aborting implementations. From this observation, there is a very simple recipe to create a forgery:

1. Make query on a message that closely resembles the message you wish to forge. The decryption of our forged message will be very similar to this message, except for 5 randomised blocks. Call this ciphertext $\overline{C}$.
2. Ask for the decryption of $\overline{C}||0$. With overwhelming probability this will fail, but due to the early–abort, the internal variables will be correct for all the blocks in the main body. This is because the scheme aborts before propagating the invalidity to the rest of the state.
3. Record the internal variable Y_i' for each of the two pairs of blocks we will randomise in our forgery (for simplicity, assume Y_1', Y_2').
4. Let $Z = Y_1' \oplus Y_2'$, and replace blocks C_i with $C_i \oplus Z$ for $i = 1, 2$ to generate a new ciphertext $\tilde{C}$.
5. Submit $\tilde{C}$ as the forgery attempt: it will be valid.

The reason $\tilde{C}$ is a valid forgery is that our replacements of C_1 and C_2 simply exchanged Y_1' and Y_2'. Thus, $Y_1 \oplus Y_2 = C_1' \oplus C_2' \oplus f^{0,0}(Y_1') \oplus f^{0,0}(Y_2')$ remains unchanged. Similarly, $X_1 \oplus X_2 =$

$Y'_1 \oplus Y'_2 \oplus f^{2,1}(S) \oplus f^{2,2}(S)$ remains constant. Therefore, the difference never propagates into X or Y, and this in turn means the difference does not leave affect any of the other pairs of blocks.

So, with the exception of the four blocks $\{M_1, M'_1, M_2, M'_2\}$, the decrypted message will be equal to that originally queried. In particular, the final two blocks M_x, M_y will remain unchanged. Since these will encode the redundancy, the forgery will be accepted. [7]

[7] Assuming $\tau \leq 2n$. If this does not hold, the forgery will still be valid as long as the modified blocks are not too near the end of the message