A preliminary version of this paper appears in CRYPTO 2019. This is a revised full version, with significant changes over the prior full version.

Nonces are Noticed: AEAD Revisited

Mihir Bellare[1] Ruth Ng[2] Björn Tackmann[3]

November 2019

Abstract

We draw attention to a gap between theory and usage of nonce-based symmetric encryption, under which the way the former treats nonces can result in violation of privacy in the latter. We bridge the gap with a new treatment of nonce-based symmetric encryption that modifies the syntax (decryption no longer takes a nonce), upgrades the security goal (asking that not just messages, but also nonces, be hidden) and gives simple, efficient schemes conforming to the new definitions. We investigate both basic security (holding when nonces are not reused) and advanced security (misuse resistance, providing best-possible guarantees when nonces are reused).

[1] Department of Computer Science & Engineering, University of California San Diego, 9500 Gilman Drive, La Jolla, California 92093, USA. Email: mihir@eng.ucsd.edu. URL: http://cseweb.ucsd.edu/~mihir/. Supported in part by NSF grants CNS-1526801 and CNS-1717640, ERC Project ERCC FP7/615074 and a gift from Microsoft.

[2] Department of Computer Science & Engineering, University of California San Diego, 9500 Gilman Drive, La Jolla, California 92093, USA. Email: ring@eng.ucsd.edu. URL: http://cseweb.ucsd.edu/~ring. Supported by DSO National Laboratories

[3] IBM Research – Zurich, Säumerstrasse 4, 8803 Rüschlikon, Switzerland. Email: bta@zurich.ibm.com. URL: https://researcher.watson.ibm.com/researcher/view.php?person=zurich-BTA.

Contents

1 Introduction

This paper revisits nonce-based symmetric encryption, raising some concerns, and then addressing them, via a new syntax, a new framework of security definitions, and schemes that offer both usability and security benefits.

BACKGROUND. As the applications and usage of symmetric encryption have evolved and grown, so has a theory that seeks to support and guide them. A definition of symmetric encryption (as with any other primitive) involves a *syntax* and then, for this syntax, definitions of *security*. In the first modern treatment [11], the syntax asked the encryption algorithm to be randomized or stateful. Security for these syntaxes evolved from asking for various forms of privacy [11] to asking for both privacy and authenticity [16, 13, 38], inaugurating authenticated encryption (AE). The idea that encryption be a deterministic algorithm taking as additional input a non-repeating quantity called a nonce seems to originate in [56] and reached its current form with Rogaway [52, 54].

NBE1 AND AE1-SECURITY. We refer to the syntax of this current form of nonce-based symmetric encryption [52, 54] as NBE1. An NBE1 scheme $\mathsf{SE1}$ specifies a *deterministic* encryption algorithm $\mathsf{SE1.Enc}$ that takes the key K, a nonce N, message M and a header (also called associated data) H to return what we call a core ciphertext C_1. Deterministic decryption algorithm $\mathsf{SE1.Dec}$ takes K, N, C_1, H to return either a message or $\perp$.

Security asks for privacy of M and integrity of both M and H *as long as nonces are unique*, meaning not re-used. Rogaway's formalization [52] asks that an adversary given oracles for encryption (taking nonce, message and header) and decryption (taking nonce, core ciphertext and header) be unable to distinguish between the case where they perform their prescribed tasks under a hidden key, and the case where the former returns random strings and the latter returns $\perp$, as long as the adversary does not repeat a nonce across its encryption queries. We will refer to this as basic AE1-security.

NBE1 providing basic AE1-security has been the goal of recent schemes, standards and proposed standards, as witnessed by GCM [45, 26] (used in TLS), OCB [56, 53, 40], CAESAR candidates [19] and RFC 5116 [44]. The security of NBE1, which we revisit, is thus of some applied interest.

THE GAP. Our concern is a gap between theory and usage that can result in privacy vulnerabilities in the latter. Recall that the decryption algorithm $\mathsf{SE1.Dec}$, to be run by the receiver, takes as input not just the key K, core ciphertext C_1 and header H, but *also the nonce* N. The theory says that how the receiver gets the nonce is "outside of the model" [52] or that it is assumed to be communicated "out-of-band" [54]. Usage cannot so dismiss it, and must find a way to convey the nonce to the receiver. The prevailing understanding, reflected in the following quote from RBBK [56], is that this is a simple matter— if the receiver does not already have the nonce N, just send it in the clear along with the core ciphertext C_1:

> The nonce N is needed both to encrypt and to decrypt. Typically it would be communicated, in the clear, along with the (core) ciphertext.

RFC 5116 is a draft standard for an interface for authenticated encryption [44]. It also considers it fine to send the nonce in the clear:

> ... there is no need to coordinate the details of the nonce format between the encrypter and the decrypter, as long *the entire nonce is sent* or stored with the ciphertext and is thus available to the decrypter ... the nonce MAY be stored *or transported* with the ciphertext ...

To repeat and summarize, the literature and proposed standards suggest transmitting what we call the "full" ciphertext, consisting of the nonce and the core ciphertext. Yet, as we now explain, this can be wrong.

NONCES CAN COMPROMISE PRIVACY. We point out that communicating a nonce in the clear with the ciphertext can damage, or even destroy, message privacy. One simple example is a nonce $N = F(M)$ that is a hash —under some public, collision-resistant hash function F— of a low-entropy message M, meaning one, like a password, which the attacker knows is likely to fall in some small set or dictionary D. Given a (full) ciphertext $C_2 = (N, C_1)$ consisting of the core ciphertext $C_1 = \mathsf{SE1.Enc}(K, N, M, H)$ together with the nonce $N = F(M)$, the attacker can recover M via "For $M' \in D$ do: If $F(M') = N$ then return M'." To take a more extreme case, consider that the nonce is some part of the message, or even the entire message, in which case the full ciphertext clearly reveals information about the message.

The concern that (adversary-visible) nonces compromise privacy, once identified, goes much further. Nonces are effectively meta-data. Even recommended and innocuous-seeming choices like counters, device identities, disk-sector numbers or packet headers reveal information about the system and identity of the sender. For example, the claim that basic-AE1-secure NBE1 provides anonymity —according to [55, Slide 19/40], this is a dividend of the requirement that core ciphertexts be indistinguishable from random strings— is moot when the nonce includes sender identity. Yet the latter is not only possible but explicitly recommended in RFC 5116 [44], which says: "When there are multiple devices performing encryption ... use a nonce format that contains a field that is distinct for each one of the devices." As another concrete example, counters are *not* a good choice of nonce from a user privacy perspective, as pointed out by Bernstein [20] and the ECRYPT-CSA *Challenges in Authenticated Encryption* report [5].

The above issues apply to all NBE1 schemes and do not contradict their (often, proven) AE1-security. They are not excluded by the unique nonce requirement or by asking for misuse resistance [57], arising in particular for the encryption of a single message with a single corresponding nonce.

A natural critique is that the privacy losses we have illustrated occur only for "pathological" choices of nonces, and choices made in practice, such as random numbers or counters, are "fine." This fails, first, to recognize the definitional gap that allows the "pathological" choices. With regard to usage, part of the selling point of NBE1 was exactly that *any* (non-repeating, unique) nonce is fine, and neither existing formalisms [52] nor existing standards [44] preclude nonce choices of the "pathological" type. Also, application designers and users cannot, and should not, carry the burden of deciding which nonces are "pathological" and which are "fine," a decision that may not be easy. (And as discussed above, for example, counters may *not* be fine.) Finally, Section 9 indicates that poor choices can in fact arise in practice.

Our perspective is that the above issues reflect a gap between the NBE1 formalism and the privacy provided by NBE1 in usage. Having pointed out this gap, we will also bridge it.

CONTRIBUTIONS IN BRIEF. The first contribution of this paper is to suggest that the way NBE1 treats nonces can result (as explained above) in compromise of privacy of messages or users. The second contribution is to address these concerns. We give a modified syntax for nonce-based encryption, called NBE2, in which decryption does not get the nonce, a corresponding framework of security definitions called AE2 that guarantee nonce privacy in addition to authenticity and message privacy, and simple ways to turn NBE1 AE1-secure schemes into NBE2 AE2-secure schemes.

AE2-secure NBE2 obviates application designers and users from the need to worry about privacy implications of their nonce choices, simplifying design and usage. With AE2-secure NBE2, one can use any nonce, even a message-dependent one such as a hash of the message, without compromising privacy of the message. And the nonces themselves are hidden just as well as messages, so user-identifying information in nonces doesn't actually identify users.

OUR NBE2 SYNTAX. In an NBE2 scheme $\mathsf{SE2}$, the inputs to the deterministic encryption algorithm $\mathsf{SE2.Enc}$ continue to be key K, nonce N, message M and header H, the output C_2 now called a ciphertext rather than a core ciphertext. The deterministic decryption algorithm $\mathsf{SE2.Dec}$ *no longer gets a nonce*, taking just key K, ciphertext C_2 and header H to return either a message M or $\perp$.

Just as an interface, NBE2 already benefits application designers and users, absolving them of the burden they had, under NBE1, of figuring out and architecting a way to communicate the nonce from sender to receiver. The NBE2 receiver, in fact, is nonce-oblivious, not needing to care, or even know, that something called a nonce was used by the sender. By reducing choice (how to communicate the nonce), NBE2 reduces error and misuse.

We associate to a given NBE1 scheme $\mathsf{SE1}$ the NBE2 scheme $\mathsf{SE2} = \mathbf{TN}[\mathsf{SE1}]$ that sets the ciphertext to the nonce plus the core ciphertext: $\mathsf{SE2.Enc}(K, N, M, H) = (N, \mathsf{SE1.Enc}(K, N, M, H))$ and $\mathsf{SE2.Dec}(K, (N, C_1), H) = \mathsf{SE1.Dec}(K, N, C_1, H)$. We refer to $\mathbf{TN}$ as the Transmit Nonce transform. This is worth defining because it will allow us, in Section 5, to formalize the above-discussed usage weaknesses in NBE1, but $\mathsf{SE2} = \mathbf{TN}[\mathsf{SE1}]$ is certainly not nonce hiding and will fail to meet the definitions we discuss next.

OUR AE2-SECURITY FRAMEWORK. Our AE2 game gives the adversary an encryption oracle ENC (taking nonce N, message M and header H to return a ciphertext C_2) and decryption oracle DEC (as per the NBE2 syntax, taking ciphertext C_2 and header H but no nonce, to return either a message M or $\perp$). When the challenge bit is $b = 1$, these oracles reply as per the encryption algorithm $\mathsf{SE2.Enc}$ and decryption algorithm $\mathsf{SE2.Dec}$ of the scheme, respectively, using a key chosen by the game. When the challenge bit is $b = 0$, oracle ENC returns a ciphertext that is drawn at random from a space $\mathsf{SE2.CS}(|N|, |M|, |H|)$ that is prescribed by the scheme $\mathsf{SE2}$ and that depends only on the lengths of the nonce, message and header, which guarantees privacy of both the nonce and message. (This space may be, but unlike for AE1 need not be, the set of all strings of some length, because NBE2 ciphertexts, unlike NBE1 core ciphertexts, may have some structure.) In the $b = 0$ case, decryption oracle DEC returns $\perp$ on any non-trivial query. The adversary eventually outputs a guess b' as to the value of b, and its advantage is $2\Pr[b = b'] - 1$.

We say that $\mathsf{SE2}$ is AE2[$\mathcal{A}$]-secure if practical adversaries in the class $\mathcal{A}$ have low advantage. Let $\mathcal{A}^{\mathrm{ae2}}_{\text{u-n}}$ be the class of unique-nonce adversaries, meaning ones that do not reuse a nonce across their ENC queries. We refer to AE2[$\mathcal{A}^{\mathrm{ae2}}_{\text{u-n}}$]-security as basic AE2-security. As the nonce-hiding analogue of basic AE1-security, it will be our first and foremost target.

Before moving to schemes, we make two remarks. First that above, for simplicity, we described our definitions in the single-user setting, but the definitions and results in the body of the paper are in the multi-user setting. Second, the framework of a single game with different notions captured via different adversary classes allows us to unify, and compactly present, many variant definitions, including basic, advanced (misuse resistance), privacy-only and random-nonce security, and in Section 3 we give such a framework not just for AE2 but also for AE1.

OUR GENERAL RESULTS. The analysis of schemes is simplified by some general results we give in Section 4. Foremost is a decomposition theorem that tightly bounds the ae-advantage of a given adversary in terms of the advantage of a privacy-only adversary (no decryption queries) and a very restricted type of authenticity adversary that we call *orderly*— it needs only verification queries (not decryption queries) and these follow its encryption queries and are all made in parallel. Here we are following Bose, Hoang and Tessaro (BHT) [22], who gave such a result for basic AE1-security. Theorem 4.1 slightly improves their bound and also extends the result to both advanced security and AE2, our single theorem thus capturing four results. Additionally, Theorem 4.2 states the standard reduction of mu security to su security and Theorem 4.3 reduces security for random

NBE2 scheme	AE2-security provided	
	Basic	Advanced
HN1[SE1, F]	Yes	Yes
HN2[SE1, ℓ, E, Spl]	Yes	Yes if $\ell \geq 128$
HN3[SE1, F]	Yes	No
HN4[SE1, ℓ, F]		Yes
HN5[TE, ℓ, ℓ_z]		Yes

Figure 1: Security attributes of the NBE2 schemes defined by our Hide-Nonce (HN) transforms. In the table SE1 denotes an NBE1 scheme, F a PRF, E a block cipher, and TE a variable-length tweakable block cipher. Spl is a splitting function, and ℓ, ℓ_z are non-negative integer parameters. A blank entry in the Basic column means the transform is not for that purpose. Note that **HN1**'s advanced security only holds when ciphertexts have sufficiently large (e.g. 128 bits) minimum length, and **HN2**'s depends on the length of the stolen ciphertext.

---

nonces to security for unique nonces.

Our transforms. In the presence of a portfolio of efficient AE1-secure NBE1 schemes supported by proofs of security with good concrete bounds [56, 45, 19, 40, 36, 60, 49, 31, 50, 30, 22, 35], designing AE2-secure NBE2 schemes from scratch seems a step backwards. Instead we give simple, cheap ways to transform AE1-secure NBE1 schemes into AE2-secure NBE2 schemes, obtaining a corresponding portfolio of AE2-secure NBE2 schemes and also allowing implementors to more easily upgrade deployed AE1-secure NBE1 to AE2-secure NBE2.

Since NBE2 schemes effectively take care of nonce communication, we expect ciphertext length to grow by at least SE1.nl, the nonce length of the base NBE1 scheme. The *ciphertext overhead* is defined as the difference between the ciphertext length and the sum of plaintext length and SE1.nl. *All our transforms have zero ciphertext overhead.* One challenge in achieving this is that nonce lengths like SE1.nl = 96 are widely-used but short of the block length 128 of many blockciphers, precluding inclusion of an extra blockcipher output in the ciphertext. With regard to computational overhead, the challenge is that it should be constant, meaning independent of the lengths of the message and header for encryption, and of the ciphertext and header for decryption. *All our transforms have constant computational overhead.*

The following discussion first considers achieving basic security and then advanced security. Security attributes of our corresponding "Hide-Nonce (HN)" transforms are summarized in Figure 1.

Basic HN transforms. We prove that all the following transforms turn a basic-AE1-secure NBE1 scheme SE1 into a basic-AE2-secure NBE2 scheme SE2. (Recall basic means nonces are unique, never reused across encryption queries.) Pseudocode and pictures for the transforms are in Figure 5.

Having first produced a core ciphertext C_1 under SE1, the idea of scheme SE2 = **HN1**[SE1, F] is to use C_1 itself as a nonce to encrypt the actual nonce in counter mode under PRF F. A drawback is that this requires the minimal core-ciphertext length SE1.mccl to be non-trivial, like at least 128, which is not true for all SE1. Scheme SE2 = **HN2**[SE1, ℓ, E, Spl] turns to the perhaps more obvious idea of enciphering the nonce with a PRF-secure blockcipher E. The difficulty is the typicality of 96-bit nonces and 128-bit blockciphers, under which naïve enciphering would add a 32-bit ciphertext overhead, which we resolve by ciphertext stealing, ℓ representing the number of stolen bits (32 in

our example) and Spl an ability to choose how the splitting is done. Scheme $\mathsf{SE2} = \mathbf{HN3}[\mathsf{SE1}, \mathsf{F}]$ uses the result of PRF F on the actual nonce as a derived nonce under which to run $\mathsf{SE1}$. This is similar to SIV [57, 49]; the difference is to achieve AE2 rather than AE1 and to apply the PRF only to the nonce (rather than nonce, message and header) to have constant computational overhead.

ADVANCED HN TRANSFORMS. Unique nonces are easier to mandate in theory than assure in practice, where nonces may repeat due to errors, system resets, or replication. In that case (returning here to NBE1), not only does basic AE1-security give no security guarantees, but also damaging attacks are possible for schemes including CCM and GCM [37, 59]. Rogaway and Shrimpton's misuse resistant NBE1, which we refer to as advanced-AE1-secure NBE1, minimizes the damage from reused nonces, retaining AE1-security as long as no nonce, message, header triple is re-encrypted [57]. This still being for the NBE1 syntax, however, the concerns with adversary-visible nonces compromising message and user privacy are unchanged. We seek the NBE2 analogue, correspondingly defining and achieving advanced-AE2-secure NBE2 to provide protection against reused nonces while also hiding them.

With our framework, the definition is easy, calling for no new games; the goal is simply $\mathrm{AE2}[\mathcal{A}^{\mathrm{ae2}}_{\text{u-nmh}}]$-security where $\mathcal{A}^{\mathrm{ae2}}_{\text{u-nmh}}$ is the class of unique-nonce, message, header adversaries, meaning ones that do not repeat a query to their ENC oracle. The presence of well-analyzed advanced-AE1-secure NBE1 schemes [57, 33, 31, 30, 22] again motivates transforms rather than from-scratch designs.

We start by revisiting our basic-security preserving transforms, asking whether they also preserve advanced security, meaning, if the starting NBE1 scheme is advanced-AE1-secure, is the transformed NBE2 scheme advanced-AE2-secure? We show that for **HN1**, the answer is YES. We then show that it is YES also for **HN2** as long as the amount ℓ of stolen ciphertext is large enough. (In practical terms, at least 128.) For **HN3**, the answer is NO.

That **HN1** and **HN2** have these properties is good, but we would like to do better. (Limitations of the above are that **HN1** puts a lower bound on $\mathsf{SE1.mccl}$ that is not always met, and setting $\ell = 128$ in **HN2** with typical 96-bit nonces will call for a 224-bit blockcipher.) We offer **HN4** and **HN5**, showing they provide advanced AE2-security. Pseudocode and pictures are in Figure 6.

Scheme $\mathsf{SE2} = \mathbf{HN4}[\mathsf{SE1}, \ell, \mathsf{F}]$ uses the result of PRF F on the actual nonce, message and header as a derived nonce for $\mathsf{SE1}$. The difference with SIV [57, 49] is that what is encrypted under $\mathsf{SE1}$ includes the actual nonce in order to hide it. The computational overhead stays constant because $\mathsf{SE1}$ need provide only privacy, which it can do in one pass. Scheme $\mathsf{SE2} = \mathbf{HN5}[\mathsf{TE}, \ell, \ell_z]$ is different, using the encode-then-encipher paradigm [16] to set the ciphertext to an enciphering, under an arbitrary-input-length, tweakable cipher TE, of the nonce, message and ℓ_t-bits of redundancy, with the header as tweak. Instantiating TE via the very fast AEZ tweakable block cipher [33] yields correspondingly fast, advanced-AE2-secure NBE2.

DEDICATED TRANSFORM FOR GCM. While our generic transforms are already able, with low overhead, to immunize GCM [45, 26] —by this we mean turn this basic-AE1-secure NBE1 scheme into a basic-AE2-secure NBE2 scheme— we ask if a dedicated transform —one that exploits the structure of GCM— can do even better. The goal is not just even lower cost overhead, but minimization of software changes. We show that simply pre-pending a block of 0s, of length equal to the nonce length, to the message, and then GCM-encrypting, provides basic-AE2-security. This means no new key materiel needs to be added, and existing encryption software can be used in a blackbox way. Ciphertext overhead remains zero. Decryption software does however need a change.

The proceedings version of our paper [14] had claimed basic-AE2-security of our GCM variant assuming the blockcipher E was prp-cca secure (also called strong prp-security, this means the

adversary is allowed both forward and backward queries) and the hash family H was AXU. In this full version, we do better, reducing the assumption on E to just PRF security, and that on H to computational AXU. The proof of security is greatly simplified by establishing privacy and authenticity separately, which suffices courtesy of our general decomposition result (Theorem 4.1). Privacy is easily reduced (Theorem 8.1) to that of GCM itself, allowing us to conclude it via known results on the latter [45, 36, 18, 43, 35] and in particular to inherit the good bounds of [35]. The proof of Theorem 8.2, establishing authenticity, is more invasive and in our view the most non-trivial proof in this paper.

Related work. In a 2013 mailing list message, Bernstein [20] argues that the security definitions for authenticated encryption fail to fully capture practical requirements, giving sequence privacy leakage via sequence-number nonces as an explicit example. AE2-secure NBE2 addresses these concerns. Bernstein also proposed a solution that can be seen as a specific instantiation of our **HN2** transformation.

As a technical step in achieving security against release of unverified plaintext (RUP), Ashur, Dunkelman and Luykx (ADL) [4] use a syntax identical to NBE2, and their techniques bear some similarities with ours that we discuss further in Section 8.

The CAESAR competition's call for authenticated encryption schemes describes a syntax where encryption receives, in place of a nonce, a public message number (PMN) and a secret message number (SMN), decryption taking only the former [23]. The formalization of Namprempre, Rogaway and Shrimpton (NRS) [48] dubs this "AE5." In this light, an NBE1 scheme is a AE5 scheme without a SMN and an NBE2 scheme is an AE5 scheme without a PMN.

Possible future work. The concerns we have raised with regard to a gap between theory and usage, and privacy vulnerabilities created by adversary-visible nonces in the latter, arise fundamentally from the choice of *syntax* represented by NBE1, and as such hold also in other contexts where an NBE1-style syntax is used. This includes AE secure under release of unverified plaintext [3], KDM-secure AE [12, 21, 24], robust AE [27], online AE [28, 34], committing AE [29, 25], indifferentiable AE [6], subtle AE [8], leakage-resilient AE [7, 21] and MiniAE [47]. A direction for future work is to treat these with an NBE2-style syntax (decryption does not get the nonce) to provide nonce hiding.

While our transforms can be applied to promote the advanced-AE1-secure AES-GCM-SIV NBE1 scheme [30] to an advanced-AE2-secure NBE2 scheme, the bounds we get are inferior to those of [22]. Bridging this gap to get advanced-AE2-secure NBE2 with security bounds like [22] is a direction for future work. Another is to prove better bounds for the authenticity of our AE2-secure version of GCM, in the vein of those for GCM [43, 35].

2 Preliminaries

Notation and terminology. By ε we denote the empty string. By $|Z|$ we denote the length of a string Z. If Z is a string then $Z[i..j]$ is bits i through j of Z if $1 \le i \le j \le |Z|$, and otherwise is ε. By $x\|y$ we denote the concatenation of strings x, y. If x, y are equal-length strings then $x \oplus y$ denotes their bitwise xor. If i is an integer then $\langle i \rangle_n \in \{0,1\}^n$ denotes the representation of $i \bmod 2^n$ as a string of (exactly) n bits. (For example, $\langle 3 \rangle_4 = 0011$.) If S is a finite set, then $|S|$ denotes it size. We say that a set S is *length-closed* if, for any $x \in S$ it is the case that $\{0,1\}^{|x|} \subseteq S$. (This will be a requirement for message, header and nonce spaces.)

If D, R are sets and $f : D \to R$ is a function then its image is $\mathrm{Im}(f) = \{\, f(x) \,:\, x \in D \,\} \subseteq R$. By $\mathrm{FUNC}(D, R)$ we denote the set of all functions $f : D \to R$. If $|D| = |R|$ then by $\mathrm{BFUNC}(D, R)$

we denote the set of all bijections $f : D \to R$. Then $\mathrm{PERM}(D) = \mathrm{BFUNC}(D, D)$ is the set of all permutations $\pi : D \to D$.

If X is a finite set, we let $x \leftarrow\!\!\$\, X$ denote picking an element of X uniformly at random and assigning it to x. Algorithms may be randomized unless otherwise indicated. If A is an algorithm, we let $y \leftarrow A^{\mathrm{O}_1,\cdots}(x_1, \ldots; \omega)$ denote running A on inputs $x_1, \ldots$ and coins ω, with oracle access to $\mathrm{O}_1, \ldots$, and assigning the output to y. By $y \leftarrow\!\!\$\, A^{\mathrm{O}_1,\cdots}(x_1, \ldots)$ we denote picking ω at random and letting $y \leftarrow A^{\mathrm{O}_1,\cdots}(x_1, \ldots; \omega)$. We let $[A^{\mathrm{O}_1,\cdots}(x_1, \ldots)]$ denote the set of all possible outputs of A when run on inputs $x_1, \ldots$ and with oracle access to $\mathrm{O}_1, \ldots$. An adversary is an algorithm. Running time is worst case, which for an algorithm with access to oracles means across all possible replies from the oracles. We use $\perp$ (bot) as a special symbol to denote rejection, and it is assumed to not be in $\{0,1\}^*$.

GAMES. We use the code-based game-playing framework of BR [17]. A game G (see Fig. 2 for an example) starts with an optional INIT procedure, followed by a non-negative number of additional procedures called oracles, and ends with a FIN procedure. Execution of adversary A with game G consists of running A with oracle access to the game procedures, with the restrictions that A's first call must be to INIT (if present), its last call must be to FIN, and it can call these procedures at most once. The output of the execution is the output of FIN. By $\Pr[\mathrm{G}(A)]$ we denote the probability that the execution of game G with adversary A results in this output being the boolean true.

Note that our adversaries have no output. The role of what in other treatments is the adversary output is, for us, played by the query to FIN.

Different games may have procedures (oracles) with the same names. If we need to disambiguate, we may write G.O to refer to oracle O of game G.

In games, integer variables, set variables boolean variables and string variables are assumed initialized, respectively, to 0, the empty set $\emptyset$, the boolean false and $\perp$.

REDUCTIONS. Proofs give reductions that take a G_2-adversary A_2 and specify (construct) a G_1-adversary A_1 that runs A_2 as a subroutine, itself responding to oracle queries of A_2. Let $\textsc{Init}, \mathrm{O1}_1, \ldots, \mathrm{O1}_{n_1}, \textsc{Fin}$ denote the oracles of G_1 and $\textsc{Init}, \mathrm{O2}_1, \ldots, \mathrm{O2}_{n_2}, \textsc{Fin}$ the oracles of G_2. Then we may write pseudocode of the form

Adversary $A_1^{\textsc{Init},\mathrm{O1}_1,\ldots,\mathrm{O1}_{n_1},\textsc{Fin}}$

$\vdots$

$A_2^{\textsc{Init}^*,\mathrm{O2}_1^*,\ldots,\mathrm{O2}_{n_2}^*,\textsc{Fin}^*}$ // Run A_2 with specified subroutines as oracles

$\vdots$

$\mathsf{procedure}\ \textsc{Init}^*$ // Subroutine simulating G_2.INIT

$\vdots$

$\mathsf{procedure}\ \mathrm{O2}_1^*(\ldots)$ // Subroutine simulating $\mathrm{G}_2.\mathrm{O2}_1$

$\vdots$

Here $\textsc{Init}^*, \mathrm{O2}_1^*, \ldots, \mathrm{O2}_{n_2}^*, \textsc{Fin}^*$ are subroutines, given in the code of A_1, that are responsible for simulating the corresponding oracles for A_2 in G_2, and will invoke A_1's oracles to do so. We adopt the convention that if a simulation is trivial, meaning $\mathrm{O2}_i^*(x)$ returns $\mathrm{O1}_j(x)$, then, in the superscripts to A_2, we simply write $\mathrm{O1}_j$ in place of $\mathrm{O2}_i^*$, and do not give code for the simulated oracle.

MULTI-USER SECURITY. There is growing recognition that security should be considered in the multi-user (mu) setting [9] rather than the traditional single-user (su) one. Our main definitions

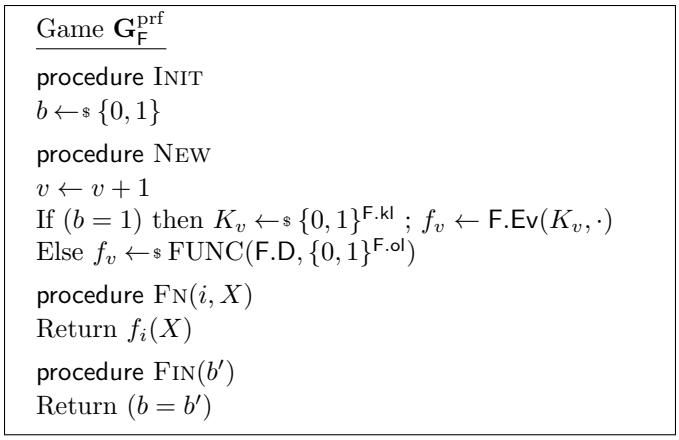

Figure 2: Game defining (multi-user) PRF security for function family F.

---

are in the mu setting. The games provide the adversary a New oracle, calling which results in a new user being initialized, with a fresh key. Other oracles are enhanced (relative to the su setting) to take an additional argument i indicating the user (key). We assume that adversaries do not make oracle queries to users (also called sessions) they have not initialized.

Function families. A function family F specifies a deterministic evaluation algorithm $\mathsf{F.Ev}: \{0,1\}^{\mathsf{F.kl}} \times \mathsf{F.D} \to \{0,1\}^{\mathsf{F.ol}}$ that takes a key K and input x to return output $\mathsf{F.Ev}(K,x)$, where $\mathsf{F.kl}$ is the key length, $\mathsf{F.D}$ is the domain and $\mathsf{F.ol}$ is the output length. We say that F is invertible if there is an inversion algorithm $\mathsf{F.In}: \{0,1\}^{\mathsf{F.kl}} \times \{0,1\}^{\mathsf{F.ol}} \to \mathsf{F.D} \cup \{\perp\}$ such that for all $K \in \{0,1\}^{\mathsf{F.kl}}$ we have (1) $\mathsf{F.In}(K,\mathsf{F.Ev}(K,x)) = x$ for all $x \in \mathsf{F.D}$, and (2) $\mathsf{F.In}(K,y) = \perp$ for all $y \notin \mathrm{Im}(\mathsf{F.Ev}(K,\cdot))$. We say that F is a permutation family if it is invertible and $\mathsf{F.D} = \{0,1\}^{\mathsf{F.ol}}$. In that case, we also refer to F as a block cipher and to $\mathsf{F.ol}$ as the block length of F, which we may denote $\mathsf{F.bl}$.

PRF security. We define (multi-user) PRF security [10] for a function family F and adversary A via the game $\mathbf{G}^{\mathrm{prf}}_{\mathsf{F}}(A)$ in Fig. 2. Here b is the challenge bit. It is required that any $\mathrm{Fn}(i,X)$ query of A satisfies $i \le v$ and $X \in \mathsf{F.D}$. The PRF advantage of adversary A is $\mathbf{Adv}^{\mathrm{prf}}_{\mathsf{F}}(A) = 2\Pr[\mathbf{G}^{\mathrm{prf}}_{\mathsf{F}}(A)] - 1$.

3 Two frameworks for nonce-based encryption

We give definitions for both AE1-secure NBE1—current nonce-based encryption [56, 52, 54]— and AE2-secure NBE2—our new nonce-based encryption. In each case there is a single security game, different variant definitions then being captured by different adversary classes. This allows a unified and compact treatment.

NBE1. An NBE1 scheme $\mathsf{SE1}$ specifies several algorithms and related quantities, as follows. Deterministic encryption algorithm $\mathsf{SE1.Enc}: \mathsf{SE1.KS} \times \mathsf{SE1.NS} \times \mathsf{SE1.MS} \times \mathsf{SE1.HS} \to \{0,1\}^*$ takes a key K in the (finite) key-space $\mathsf{SE1.KS}$, a nonce N in the nonce-space $\mathsf{SE1.NS}$, a message M in the message space $\mathsf{SE1.MS}$ and a header H in the header space $\mathsf{SE1.HS}$ to return what we call a core ciphertext C_1. This is a string of length $\mathsf{SE1.ccl}(|N|,|M|,|H|)$, where $\mathsf{SE1.ccl}$ is the core-ciphertext length function. $\mathsf{SE1}$ also specifies a deterministic decryption algorithm $\mathsf{SE1.Dec}: \mathsf{SE1.KS} \times \mathsf{SE1.NS} \times \{0,1\}^* \times \mathsf{SE1.HS} \to \mathsf{SE1.MS} \cup \{\perp\}$ that takes key K, nonce N, core ciphertext C_1 and header H to return an output that is either a message $M \in \mathsf{SE1.MS}$, or $\perp$. It is required

Game $\mathbf{G}^{\mathrm{ae1}}_{\mathsf{SE1}}$	Game $\mathbf{G}^{\mathrm{ae2}}_{\mathsf{SE2}}$
procedure INIT $b \leftarrow\!\!\$\ \{0,1\}$	procedure INIT $b \leftarrow\!\!\$\ \{0,1\}$
procedure NEW $v \leftarrow v+1$; $K_v \leftarrow\!\!\$\ \mathsf{SE1.KS}$	procedure NEW $v \leftarrow v+1$; $K_v \leftarrow\!\!\$\ \mathsf{SE2.KS}$

procedure ENC(i,N,M,H)
If $(b=1)$ then
  $C_1 \leftarrow \mathsf{SE1.Enc}(K_i,N,M,H)$
  Else $C_1 \leftarrow\!\!\$\ \{0,1\}^{\mathsf{SE1.ccl}(\lvert N\rvert,\lvert M\rvert,\lvert H\rvert)}$
$\mathbf{M}[i,N,C_1,H] \leftarrow M$; Return C_1
	procedure ENC(i,N,M,H)
	If $(b=1)$ then
	  $C_2 \leftarrow \mathsf{SE2.Enc}(K_i,N,M,H)$
	  Else $C_2 \leftarrow\!\!\$\ \mathsf{SE2.CS}(\lvert N\rvert,\lvert M\rvert,\lvert H\rvert)$
	$\mathbf{M}[i,C_2,H] \leftarrow M$; Return C_2

procedure DEC(i,N,C_1,H)
If $(\mathbf{M}[i,N,C_1,H] \neq \perp)$ then
  Return $\mathbf{M}[i,N,C_1,H]$
If $(b=0)$ then $M \leftarrow \perp$
  Else $M \leftarrow \mathsf{SE1.Dec}(K_i,N,C_1,H)$
Return M
	procedure DEC(i,C_2,H)
	If $(\mathbf{M}[i,C_2,H] \neq \perp)$ then
	  Return $\mathbf{M}[i,C_2,H]$
	If $(b=0)$ then $M \leftarrow \perp$
	  Else $M \leftarrow \mathsf{SE2.Dec}(K_i,C_2,H)$
	Return M

procedure FIN(b') Return $(b=b')$	procedure FIN(b') Return $(b=b')$

$\mathcal{A}^{\mathrm{x}}_{\text{u-n}}$	Unique nonce adversaries — $A \in \mathcal{A}^{\mathrm{x}}_{\text{u-n}}$ does not repeat a user-nonce pair i,N across its ENC queries
$\mathcal{A}^{\mathrm{x}}_{\text{u-nmh}}$	Unique nonce-message-header adversaries — $A \in \mathcal{A}^{\mathrm{x}}_{\text{u-nmh}}$ does not repeat a query to ENC
$\mathcal{A}^{\mathrm{x}}_{\text{priv}}$	Privacy adversaries — $A \in \mathcal{A}^{\mathrm{x}}_{\text{priv}}$ makes no DEC queries
$\mathcal{A}^{\mathrm{x}}_{1}$	Single-user adversaries — $A \in \mathcal{A}^{\mathrm{x}}_{1}$ makes only one NEW query
$\mathcal{A}^{\mathrm{x}}_{\text{r-n}}$	Random-nonce adversaries — The nonces in the ENC queries of $A \in \mathcal{A}^{\mathrm{x}}_{\text{r-n}}$ are distributed uniformly and independently at random

Figure 3: Game defining AE1-security of NBE1 scheme $\mathsf{SE1}$ (top left), game defining AE2-security of NBE2 scheme $\mathsf{SE2}$ (top right), and some classes of adversaries, leading to different security notions, where $\mathrm{x} \in \{\mathrm{ae1}, \mathrm{ae2}\}$ (bottom).

---

that $\mathsf{SE1.NS}, \mathsf{SE1.MS}, \mathsf{SE1.HS}$ are length-closed sets as defined in Section 2. Most often nonces are of a fixed length denoted $\mathsf{SE1.nl}$, meaning $\mathsf{SE1.NS} = \{0,1\}^{\mathsf{SE1.nl}}$. Decryption correctness requires that $\mathsf{SE1.Dec}(K,N,\mathsf{SE1.Enc}(K,N,M,H),H) = M$ for all $K \in \mathsf{SE1.KS}$, $N \in \mathsf{SE1.NS}$, $M \in \mathsf{SE1.MS}$ and $H \in \mathsf{SE1.HS}$.

AE1 GAME AND ADVANTAGE. Let $\mathsf{SE1}$ be an NBE1 scheme and A an adversary. We associate to them the game $\mathbf{G}^{\mathrm{ae1}}_{\mathsf{SE1}}(A)$ shown on the top left of Fig. 3. (We use the name "AE1" to associate the game with the NBE1 syntax). The AE1-advantage of adversary A is $\mathbf{Adv}^{\mathrm{ae1}}_{\mathsf{SE1}}(A) = 2\Pr[\mathbf{G}^{\mathrm{ae1}}_{\mathsf{SE1}}(A)] - 1$. The game is in the multi-user setting, oracle NEW allowing the adversary to initialize a new user with a fresh key. It is required that any $\mathrm{ENC}(i,N,M,H)$ query of A satisfy $1 \leq i \leq v$, N

$\in \mathsf{SE1.NS}$, $M \in \mathsf{SE1.MS}$ and $H \in \mathsf{SE1.HS}$. When the challenge bit b is 1, the encryption oracle will return a core ciphertext as stipulated by $\mathsf{SE1.Enc}$, using the key for the indicated user i. In the $b = 0$ case, ENC will return a random string of length $\mathsf{SE1.ccl}(|N|, |M|, |H|)$. The array $\mathbf{M}$ is assumed to initially be $\perp$ everywhere, and holds core ciphertexts returned by ENC. It is required that any $\text{DEC}(i, N, C_1, H)$ query of A satisfy $1 \leq i \leq v$, $N \in \mathsf{SE1.NS}$ and $H \in \mathsf{SE1.HS}$. When the challenge bit b is 1, the decryption oracle will perform decryption as stipulated by $\mathsf{SE1.Dec}$, using the key for the indicated user i. In the $b = 0$ case, DEC will return $\perp$ on any core ciphertext not previously returned by the encryption oracle.

AE1 SECURITY METRICS. AE1-security is clearly not achievable without restrictions on the adversary. For example, if A repeats a query i, N, M, H to ENC, then, when $b = 1$ it gets back the same reply both times, while if $b = 0$ it likely does not, allowing it to determine b with high probability. We define different classes of adversaries, summarized by the table at the bottom of Figure 3, with the superscript "x" here being ae1. We say that NBE1 scheme $\mathsf{SE1}$ is AE1[$\mathcal{A}$]-secure if adversaries in $\mathcal{A}$ have low AE1-advantage. The definition is in the multi-user setting, but restricting attention to adversaries in the class $\mathcal{A}_1^{\text{ae1}}$ allows us to recover the single-user setting. Different security notions in the literature are then captured as AE1[$\mathcal{A}$]-security for different classes of adversaries $\mathcal{A}$, as we illustrate below:

- $\mathcal{A}_{\text{u-n}}^{\text{ae1}}$ is the class of adversaries whose ENC queries never repeat a user-nonce pair. AE1[$\mathcal{A}_{\text{u-n}}^{\text{ae1}} \cap \mathcal{A}_1^{\text{ae1}}$]-security is thus AEAD as defined in [52, 54].
- AE1[$\mathcal{A}_{\text{u-n}}^{\text{ae1}}$]-security is the extension of this to the multi-user setting as defined in [18], which we have referred to as basic AE1-security in Section 1.
- Adversaries in $\mathcal{A}_{\text{u-nmh}}^{\text{ae1}} \supseteq \mathcal{A}_{\text{u-n}}^{\text{ae1}}$ are allowed to re-use a user-nonce pair across ENC queries as long as they never repeat an entire query. AE1[$\mathcal{A}_{\text{u-nmh}}^{\text{ae1}} \cap \mathcal{A}_1^{\text{ae1}}$]-security is misuse resistant AE [57].
- AE1[$\mathcal{A}_{\text{u-nmh}}^{\text{ae1}}$]-security is the extension of this to the multi-user setting [22], which we have referred to as advanced-AE1-security in Section 1.
- Adversaries in $\mathcal{A}_{\text{r-n}}^{\text{ae1}}$ pick the nonces in their ENC queries uniformly and independently at random from $\mathsf{SE1.NS}$. (While the intent here is likely understandable, what precisely it means for an adversary to be in this class does actually need a careful definition, which is given in Appendix A.) No restriction is placed on how the adversary picks nonces in DEC queries. AE1[$\mathcal{A}_{\text{r-n}}^{\text{ae1}} \cap \mathcal{A}_1^{\text{ae1}}$]-security is thus classical randomized AE [13] for schemes which make encryption randomness public, which is the norm.
- Sometimes, in the unique-nonce setting, we consider schemes that provide only privacy, not authenticity, and, rather than giving a separate game, can capture this as AE1[$\mathcal{A}_{\text{priv}}^{\text{ae1}} \cap \mathcal{A}_{\text{u-n}}^{\text{ae1}}$]-security. AE1[$\mathcal{A}_{\text{priv}}^{\text{ae1}} \cap \mathcal{A}_{\text{u-n}}^{\text{ae1}} \cap \mathcal{A}_1^{\text{ae1}}$]-security is $\mathsf{IND\$\text{-}CPA}$ security, as defined in [52].

Further adversary classes can be defined to capture limited nonce reuse [22] or other resource restrictions.

We believe our (above) AE1 framework (single game, many adversary classes) is of independent interest, as a way to unify, better understand and compactly present existing and new notions of security for NBE1 schemes. We give a similar framework for AE2 next.

NBE2 SYNTAX. An NBE2 scheme $\mathsf{SE2}$ specifies several algorithms and related quantities, as follows. Deterministic encryption algorithm $\mathsf{SE2.Enc} : \mathsf{SE2.KS} \times \mathsf{SE2.NS} \times \mathsf{SE2.MS} \times \mathsf{SE2.HS} \to \{0,1\}^*$, just like for NBE1, takes a key K in the (finite) key-space $\mathsf{SE2.KS}$, a nonce N in the nonce-space $\mathsf{SE2.NS}$, a message M in the message space $\mathsf{SE2.MS}$ and a header H in the header space $\mathsf{SE2.HS}$ to return a ciphertext C_2 that is in the ciphertext space $\mathsf{SE2.CS}(|N|, |M|, |H|)$. $\mathsf{SE2}$ also specifies a deterministic decryption algorithm $\mathsf{SE2.Dec} : \mathsf{SE2.KS} \times \{0,1\}^* \times \mathsf{SE2.HS} \to \mathsf{SE2.MS} \cup \{\perp\}$ that takes

key K, ciphertext C_2 and header H to return an output that is either a message $M \in \mathsf{SE2.MS}$, or $\perp$. (Unlike in NBE1, it does *not* take a nonce input.) It is required that $\mathsf{SE2.NS}, \mathsf{SE2.MS}, \mathsf{SE2.HS}$ are length-closed sets as defined in Section 2. Most often nonces are of a fixed length denoted $\mathsf{SE2.nl}$, meaning $\mathsf{SE2.NS} = \{0,1\}^{\mathsf{SE2.nl}}$. Decryption correctness requires that $\mathsf{SE2.Dec}(K, \mathsf{SE2.Enc}(K, N, M, H), H) = M$ for all $K \in \mathsf{SE2.KS}, N \in \mathsf{SE2.NS}, M \in \mathsf{SE2.MS}$ and $H \in \mathsf{SE2.HS}$.

AE2 GAME AND ADVANTAGE. Let $\mathsf{SE2}$ be an NBE2 scheme and A an adversary. We associate to them the game $\mathbf{G}^{\mathrm{ae2}}_{\mathsf{SE2}}(A)$ shown on the top right of Fig. 3. (We use the name "AE2" to associate the game with the NBE2 syntax). The AE2-advantage of adversary A is $\mathbf{Adv}^{\mathrm{ae2}}_{\mathsf{SE2}}(A) = 2\Pr[\mathbf{G}^{\mathrm{ae2}}_{\mathsf{SE2}}(A)] - 1$. The game is in the multi-user setting, oracle NEW allowing the adversary to initialize a new user with a fresh key. It is required that any $\textsc{Enc}(i, N, M, H)$ query of A satisfy $1 \le i \le v$, $N \in \mathsf{SE2.NS}$, $M \in \mathsf{SE2.MS}$ and $H \in \mathsf{SE2.HS}$. When the challenge bit b is 1, the encryption oracle will return a ciphertext as stipulated by $\mathsf{SE2.Enc}$, using the key for the indicated user i. When $b = 0$, ENC will return a random element of the ciphertext space $\mathsf{SE2.CS}(|N|, |M|, |H|)$. The array $\mathbf{M}$ is assumed to initially be $\perp$ everywhere, and holds ciphertexts returned by ENC. It is required that any $\textsc{Dec}(i, C_2, H)$ query of A satisfy $1 \le i \le v$ and $H \in \mathsf{SE2.HS}$. When the challenge bit b is 1, the decryption oracle will perform decryption as stipulated by $\mathsf{SE2.Dec}$, using the key for the indicated user i. When $b = 0$, DEC will return $\perp$ on any ciphertext not previously returned by the encryption oracle.

AE2 SECURITY METRICS. As with AE1-security, restrictions must be placed on the adversary to achieve AE2-security, and we use adversary classes to capture restrictions corresponding to different notions of interest. The classes are summarized by the table at the bottom of Figure 3, with the superscript "x" now being ae2. The classes and resulting notions are analogous to those for AE1. Thus, AE2$[\mathcal{A}^{\mathrm{ae2}}_1]$-security recovers the single-user setting. $\mathcal{A}^{\mathrm{ae2}}_{\text{u-n}}$ is the class of adversaries whose ENC queries never repeat a user-nonce pair, so AE2$[\mathcal{A}^{\mathrm{ae2}}_{\text{u-n}}]$-security is what we have referred to as basic AE2-security in Section 1. Adversaries in $\mathcal{A}^{\mathrm{ae2}}_{\text{u-nmh}} \supseteq \mathcal{A}^{\mathrm{ae2}}_{\text{u-n}}$ are allowed to re-use a user-nonce pair across ENC queries as long as they never repeat an entire query, so AE2$[\mathcal{A}^{\mathrm{ae2}}_{\text{u-nmh}}]$-security is what we have referred to as advanced AE2-security in Section 1. Adversaries in $\mathcal{A}^{\mathrm{ae2}}_{\text{r-n}}$ pick the nonces in their ENC queries uniformly and independently at random from $\mathsf{SE2.NS}$. AE2$[\mathcal{A}^{\mathrm{ae2}}_{\mathrm{priv}}]$-security is privacy only.

DISCUSSION. The main (small but important) change in the syntax from NBE1 to NBE2 is that in the latter, the decryption algorithm no longer gets the nonce as input. It is up to encryption to ensure that the ciphertext contains everything (beyond key and header) needed to decrypt. Nonces are thus no longer magically communicated, making the interface, and the task of application designers, simpler and less error-prone, reducing the possibility of loss of privacy from poor choices of nonces and opening the door to nonce-hiding security as captured by AE2. Another change is that, rather than a ciphertext length function, an NBE2 scheme specifies a ciphertext space. The reason is that a ciphertext might have some structure, like being a pair (C, C'). Ciphertexts like this cannot be indistinguishable from random strings, but they can be indistinguishable from pairs of random strings, which is captured by defining the ciphertext space correspondingly. This follows [29], in whose committing AE definition the same issue arose.

NONCE-RECOVERING NBE2. A natural subclass of NBE2 schemes are those which recover the nonce explicitly during decryption. We provide definitions to capture such schemes. We say that an NBE2 scheme $\mathsf{SE2}$ is nonce-recovering if there exists a deterministic nonce-plus-message recovery algorithm $\mathsf{SE2.NMR}$ such that for any $(K, C_2, H) \in \mathsf{SE2.KS} \times \{0,1\}^* \times \mathsf{SE2.HS}$, if $\mathsf{SE2.NMR}(K, C_2, H) \neq \perp$ then it parses as a pair $(M, N) \in \mathsf{SE2.MS} \times \mathsf{SE2.NS}$ satisfying $\mathsf{SE2.Dec}(K, C_2, H) = M$ and

Game $\mathbf{G}^{\text{auth1}}_{\mathsf{SE1}}$	Game $\mathbf{G}^{\text{auth2}}_{\mathsf{SE2}}$
procedure NEW $v \leftarrow v + 1$; $K_v \leftarrow\!\!\$\ \mathsf{SE1.KS}$	procedure NEW $v \leftarrow v + 1$; $K_v \leftarrow\!\!\$\ \mathsf{SE2.KS}$
procedure ENC(i, N, M, H) $C_1 \leftarrow \mathsf{SE1.Enc}(K_i, N, M, H)$ $\mathcal{S} \leftarrow \mathcal{S} \cup \{(i, N, C_1, H)\}$; Return C_1	procedure ENC(i, N, M, H) $C_2 \leftarrow \mathsf{SE2.Enc}(K_i, N, M, H)$ $\mathcal{S} \leftarrow \mathcal{S} \cup \{(i, C_2, H)\}$; Return C_2

procedure VF(i, N, C_1, H)
$M \leftarrow \mathsf{SE1.Dec}(K_i, N, C_1, H)$
If $(M \neq \bot) \wedge ((i, N, C_1, H) \notin \mathcal{S})$ then
  $\mathsf{win} \leftarrow \mathsf{true}$
Return $(M = \bot)$
	procedure VF(i, C_2, H)
	$M \leftarrow \mathsf{SE2.Dec}(K_i, C_2, H)$
	If $(M \neq \bot) \wedge ((i, C_2, H) \notin \mathcal{S})$ then
	  $\mathsf{win} \leftarrow \mathsf{true}$
	Return $(M = \bot)$

procedure FIN Return win	procedure FIN Return win

Figure 4: Games defining authenticity of NBE1 scheme SE1 (left) and NBE2 scheme SE2 (right).

$\mathsf{SE2.Enc}(K, N, M, H) = C_2$. Most of our transforms from NBE1 scheme to NBE2 schemes yield nonce-recovering NBE2 schemes.

4 Some general results

We give a few general results that we will use.

PRIV+AUTH IMPLIES AE. Early definitions of authenticated encryption gave separate privacy and authenticity requirements [13]. Above, in the style of [52], a single game captures a joint privacy-and-authenticity requirement. Bose, Hoang and Tessaro (BHT) [22] showed that, for basic-secure AE1, separate, privacy-alone and authenticity alone conditions imply the joint condition. Here we extend this to both advanced security and AE2. This is useful because (1) It can be easier to establish the simpler, separate requirements than the joint one, and (2) Proven bounds can differ for privacy and authenticity, which is not visible if one only gives results for the joint notion.

Proceeding, the definition for privacy alone is already present, obtained above by restricting to adversaries in the classes AE1$[\mathcal{A}^{\text{ae1}}_{\text{priv}}]$ (for NBE1) or AE2$[\mathcal{A}^{\text{ae2}}_{\text{priv}}]$ (for NBE2). To define authenticity-alone, consider the games $\mathbf{G}^{\text{auth1}}_{\mathsf{SE1}}$ and $\mathbf{G}^{\text{auth2}}_{\mathsf{SE2}}$ in Fig. 4, where SE1 is a NBE1 scheme and SE2 is an NBE2 scheme. The auth1-advantage of adversary A is $\mathbf{Adv}^{\text{auth1}}_{\mathsf{SE1}}(A) = \Pr[\mathbf{G}^{\text{auth1}}_{\mathsf{SE1}}(A)]$. The auth2-advantage of adversary A is $\mathbf{Adv}^{\text{auth2}}_{\mathsf{SE2}}(A) = \Pr[\mathbf{G}^{\text{auth2}}_{\mathsf{SE2}}(A)]$.

As for AE, different notions of security are captured by considering different classes of adversaries. For $\mathrm{x} \in \{\text{auth1}, \text{auth2}\}$ we define:

- $\mathcal{A}^{\mathrm{x}}_{\text{u-n}}$ is the class of adversaries whose ENC queries never repeat a user-nonce pair.
- Adversaries in $\mathcal{A}^{\mathrm{x}}_{\text{u-nmh}} \supseteq \mathcal{A}^{\mathrm{x}}_{\text{u-n}}$ are allowed to re-use a user-nonce pair across ENC queries as long as they never repeat an entire query.
- Adversaries in $\mathcal{A}^{\mathrm{x}}_{\text{r-n}}$ pick the nonces in their ENC queries uniformly and independently at random from the nonce space of the scheme. This is defined in more detail in Appendix A.
- $\mathcal{A}^{\mathrm{x}}_{\text{ord}}$ is the class of adversaries that are *orderly*. An adversary is orderly if two conditions hold. First, it makes its VF queries after all its ENC queries. (That is, once it has made a VF query,

it does not make any more ENC queries.) Second, the VF queries are non-adaptive, meaning a VF query does not depend on the answer to a prior VF query. (But the VF queries can depend on answers to the prior ENC queries). Intuitively, think of an orderly adversary as first making a bunch of ENC queries, and then a bunch of VF queries in parallel.

The following theorem says that AE-security decomposes into privacy plus authenticity. The statement covers AE1 and AE2 (via the choice of X) and basic and advanced (via the choice of y) security, so that the single statement encompasses four results.

BHT [22] give and prove this result for basic AE1 secure NBE1. Our bound is slightly better than theirs, dropping an added term, and we generalize to AE2 and advanced security. As with BHT [22], the theorem allows us to restrict attention to orderly authenticity adversaries, which later makes proving authenticity simpler. The proof is in Appendix B.

Theorem 4.1 *Let* $\mathrm{X} \in \{1, 2\}$ *and* $\mathrm{y} \in \{\mathrm{n}, \mathrm{nmh}\}$. *Let* SE *be a NBEX scheme. Let* $A \in \mathcal{A}^{\mathrm{aeX}}_{\text{u-y}}$ *be an adversary. Then, we can construct adversaries* $B \in \mathcal{A}^{\mathrm{aeX}}_{\mathrm{priv}} \cap \mathcal{A}^{\mathrm{aeX}}_{\text{u-y}}$ *and* $C \in \mathcal{A}^{\mathrm{authX}}_{\mathrm{ord}} \cap \mathcal{A}^{\mathrm{authX}}_{\text{u-y}}$ *such that:*

$$\mathbf{Adv}^{\mathrm{aeX}}_{\mathsf{SE}}(A) \leq \mathbf{Adv}^{\mathrm{aeX}}_{\mathsf{SE}}(B) + \mathbf{Adv}^{\mathrm{authX}}_{\mathsf{SE}}(C) \ .$$

Adversary B *preserves the resources of* A. *Adversary* C *is orderly. Additionally, it preserves query resources to* NEW, ENC, *its queries to* VF *are those that* A *makes to* DEC, *and it preserves running time.*

FROM SINGLE- TO MULTI-USER SECURITY. The usual hybrid argument can be used to show that single-user security implies multi-user security up to a factor q_n degradation in advantage where q_n is the number of users, meaning the number of NEW queries of the adversary. As much as possible we will not rely on this but rather treat multi-user security directly, so as to avoid the degradation in the bound, but in some cases it will be easier to treat single-user security and take the hit in the bound. Accordingly we state the result here. We omit the proof since it is standard.

Theorem 4.2 *Let* $\mathrm{X} \in \{1, 2\}$ *and* $\mathrm{y} \in \{\mathrm{n}, \mathrm{nmh}\}$. *Let* SE *be a NBEX scheme. Let* $A \in \mathcal{A}^{\mathrm{aeX}}_{\text{u-y}}$ *be an adversary making* q_n *calls to its* NEW *oracle and* q_e, q_d *calls per user to its* ENC *and* DEC *oracles, respectively. Then, we can construct adversary* $A \in \mathcal{A}^{\mathrm{aeX}}_{1} \cap \mathcal{A}^{\mathrm{aeX}}_{\text{u-y}}$ *such that:*

$$\mathbf{Adv}^{\mathrm{aeX}}_{\mathsf{SE}}(A) \leq q_n \cdot \mathbf{Adv}^{\mathrm{aeX}}_{\mathsf{SE}}(B) \ .$$

Adversary B *makes* 1 *query to its* NEW *oracle and* q_e, q_d *queries to its* ENC, DEC *oracles, respectively.*

SECURITY UNDER RANDOM NONCES. The following says that AE2$[\mathcal{A}^{\mathrm{ae2}}_{\text{u-n}}]$-security (resp. AE1$[\mathcal{A}^{\mathrm{ae1}}_{\text{u-n}}]$) implies AE1$[\mathcal{A}^{\mathrm{ae1}}_{\text{r-n}}]$-security (resp. AE1$[\mathcal{A}^{\mathrm{ae1}}_{\text{r-n}}]$) with a degradation in advantage corresponding to the probability that a nonce repeats for some user. We will refer to this later. We omit the (obvious) proof.

Theorem 4.3 *Let* $\mathrm{X} \in \{1, 2\}$. *Let* SE *be a NBEX scheme. Let* $A_{\mathrm{rn}} \in \mathcal{A}^{\mathrm{aeX}}_{\text{r-n}}$ *be an adversary making* q_n *calls to its* NEW *oracle and* q_e *calls per user to its* ENC *oracle. Then, we can construct adversary* $A_{\mathrm{un}} \in \mathcal{A}^{\mathrm{aeX}}_{\text{u-n}}$ *such that*

$$\mathbf{Adv}^{\mathrm{aeX}}_{\mathsf{SE}}(A_{\mathrm{rn}}) \leq \mathbf{Adv}^{\mathrm{aeX}}_{\mathsf{SE}}(A_{\mathrm{un}}) + \frac{q_n q_e (q_e - 1)}{2^{\mathsf{SE.nl}}} \ .$$

Adversary A_{un} *preserves the resources of* A_{rn}.

Saying A_{un} preserves the resources of A_{rn} means that the number of queries to all oracles are the same for both, as is the running time.

5 Usage of NBE1: The Transmit-Nonce transform

With AE1-secure NBE1, the nonce is needed for decryption. But how does the decryptor get it? This is a question about usage not addressed in the formalism. The understanding, however, is that the nonce can be communicated in the clear, with the core ciphertext. One might argue this is fine because, in the AE1-formalism, the adversary picks the nonce, so seeing the nonce again in the ciphertext cannot give the adversary an advantage.

We have discussed in the introduction why this fails to model cases where the nonce is chosen by the user, and why, at least in general, nonce transmission may violate message privacy. But the claim, so far, was informal. The reason was that transmitting the nonce represents a *usage* of NBE1 and we had no definitions to capture this. With AE2-secure NBE2, that gap is filled and we are in a position to formalize the claim of usage insecurity.

Some readers may see this is unnecessary, belaboring an obvious point. Indeed, the intuition is clear enough. But formalizing it serves also as an introduction to exercising our framework. We capture the usage in question as an NBE2 scheme $\mathsf{SE}_{\text{TN}} = \mathbf{TN}[\mathsf{SE1}]$ built from a given NBE1 scheme $\mathsf{SE1}$ by what we call the transmit-nonce transform $\mathbf{TN}$. We detail the (rather obvious) claim that SE_{TN} fails to meet AE2-security, and discuss how it will also fail to meet other, weaker privacy goals.

The **TN** transform. Our **TN** (Transmit Nonce) transform takes an NBE1 scheme $\mathsf{SE1}$ and returns the NBE2 scheme $\mathsf{SE}_{\text{TN}} = \mathbf{TN}[\mathsf{SE1}]$, that, as the name suggests, transmits the nonce in the clear, meaning the SE_{TN} ciphertext is the nonce together with the $\mathsf{SE1}$ core ciphertext. In more detail, encryption algorithm $\mathsf{SE}_{\text{TN}}.\mathsf{Enc}(K, N, M, H)$ lets $C_1 \leftarrow \mathsf{SE1}.\mathsf{Enc}(K, N, M, H)$ and returns ciphertext $C_2 \leftarrow (N, C_1)$. Decryption algorithm $\mathsf{SE}_{\text{TN}}.\mathsf{Dec}(K, C_2, H)$ parses C_2 as a pair (N, C_1) with $N \in \mathsf{SE1.NS}$ —we write this as $(N, C_1) \leftarrow C_2$— returning $\perp$ if the parsing fails, and else returning $M \leftarrow \mathsf{SE1}.\mathsf{Dec}(K, N, C_1, H)$. NBE2 scheme SE_{TN} has the same key space, message space and header space as $\mathsf{SE1}$, and we define its ciphertext space via $\mathsf{SE}_{\text{TN}}.\mathsf{CS}(\ell_n, \ell_m, \ell_h) = \mathsf{SE1.NS} \times \{0,1\}^{\mathsf{SE1.ccl}(\ell_n, \ell_m, \ell_h)}$ for all $\ell_n, \ell_m, \ell_h \geq 0$. Usage of $\mathsf{SE1}$ in which the nonce is sent in the clear (along with the core ciphertext) can now be formally modeled by asking what formal security notions for NBE2 schemes are met by $\mathsf{SE}_{\text{TN}} = \mathbf{TN}[\mathsf{SE1}]$.

Insecurity of $\mathbf{TN}[\mathsf{SE1}]$. Let $\mathsf{SE1}$ be *any* NBE1 scheme. It might, like GCM, be $\text{AE1}[\mathcal{A}^{\text{ae1}}_{\text{u-n}}]$-secure, or it might even be $\text{AE1}[\mathcal{A}^{\text{ae1}}_{\text{u-nmh}}]$-secure. Regardless, we claim that NBE2 scheme $\mathsf{SE}_{\text{TN}} = \mathbf{TN}[\mathsf{SE1}]$ fails to be $\text{AE2}[\mathcal{A}^{\text{ae2}}_{\text{priv}} \cap \mathcal{A}^{\text{ae2}}_{\text{u-n}}]$-secure, meaning fails to provide privacy even for adversaries that do not reuse a nonce. This is quite obvious, since the adversary can test whether the nonce in its Enc query matches the one returned in the ciphertext. In detail:

Adversary $A^{\text{New},\text{Enc}}$
Init
Pick some $(N, M, H) \in \mathsf{SE1.NS} \times \mathsf{SE1.MS} \times \mathsf{SE1.HS}$ with $|N| \geq 1$
New // Initialize one user
$(N^*, C_1) \leftarrow\!\!\$\ \text{Enc}(1, N, M, H)$ // Ciphertext returned is a pair
If $(N^* = N)$ then $b' \leftarrow 1$ else $b' \leftarrow 0$
Fin(b')

This adversary has advantage $\mathbf{Adv}^{\mathrm{ae2}}_{\mathsf{SE}_{\mathrm{TN}}}(A) \geq 1-1/2 = 1/2$, so represents a violation of AE2[$\mathcal{A}^{\mathrm{ae2}}_{\mathrm{priv}} \cap \mathcal{A}^{\mathrm{ae2}}_{\mathrm{u\text{-}n}}$]-security.

Discussion. The attack above may be difficult to reconcile with $\mathsf{SE1}$ being AE1[$\mathcal{A}^{\mathrm{ae1}}_{\mathrm{u\text{-}n}}$]-secure, the question being that, in the AE1 game, the adversary picks the nonce, and thus already knows it, so why should seeing it again in the ciphertext give the adversary extra information? The answer is that in usage the adversary does not know the nonce *a priori* and seeing may provide additional information. This is not modeled in AE1 but is modeled in AE2. To be clear, the above violation of AE2 security does *not* contradict the assumed AE1-security of $\mathsf{SE1}$.

One might (correctly) argue that AE2 is a strong requirement so failing it does not represent a concerning violation of security, but it is clear that $\mathsf{SE}_{\mathrm{TN}}$ will fail to meet even much weaker notions of privacy for NBE2 schemes that one could formalize in natural ways, such as message recovery security or semantic security. (The nonce could be message dependent, in the extreme equal to the message.) One might also suggest that the losses of privacy occur for pathological choices of nonces, and nonce transmission is just fine if the nonce is a random number or counter, to which there are two responses. (1) The pitch and promise of AE1[$\mathcal{A}^{\mathrm{ae1}}_{\mathrm{u\text{-}n}}$]-secure NBE1 is that *any* (non-repeating) nonce is fine. For example RBBK [56] says "The entity that encrypts chooses a new nonce for every message with the *only* restriction that no nonce is used twice," and RFC 5116 says "Applications SHOULD use the nonce formation method defined in Section 3.2, and MAY use any other method that meets the uniqueness requirement." It is important to know (both to prevent misuse and for our understanding) that in usage of NBE1, security requires more than just uniqueness of nonces; one must be concerned with how they are conveyed to the receiver. (2) A counter nonce can lead to loss of user privacy, for example revealing identity information, that is resolved by moving to AE2[$\mathcal{A}^{\mathrm{ae2}}_{\mathrm{u\text{-}n}}$]-secure NBE2, which is nonce hiding.

Privacy violations of the type discussed here, and captured by **TN**, occur only when the nonce is transmitted in the clear. They do not arise in TLS, where the nonce is not transmitted. (It is a counter that is held, and locally updated, by both sender and receiver.)

6 Basic transforms

We have explained that AE2-secure NBE2 offers valuable security and usability benefits over current encryption. So we now turn to achieving it. We follow the development path of NBE1, first, in this section, targeting basic AE2-security —no user reuses a nonce, which in our framework corresponds to adversaries in the class $\mathcal{A}^{\mathrm{ae2}}_{\mathrm{u\text{-}n}}$— and then, in Section 7, targeting advanced AE2-security —misuse resistance, where nonce-reuse is allowed, which in our framework corresponds to adversaries in the class $\mathcal{A}^{\mathrm{ae2}}_{\mathrm{u\text{-}nmh}}$.

Significant effort has gone into the design and analysis of basic-AE1-secure NBE1 schemes. We want to leverage rather than discard this. Accordingly, rather than from-scratch designs, we seek *transforms* of basic-AE1-secure NBE1 schemes into basic-AE2-secure NBE2 ones. This section gives three transforms that are simple and efficient and minimize quantitative security loss.

6.1 Preliminaries

We assume for simplicity that the NBE1 schemes provided as input to our transforms have nonces of a fixed length, meaning that $\mathsf{SE1.NS} = \{0,1\}^{\mathsf{SE1.nl}}$. This holds for most real-world AE1-secure NBE1 schemes. All our transforms can be adapted to allow variable-length nonces.

Core ciphertexts in practical NBE1 schemes tend to be no shorter than a certain minimal value, for example 96 bits for typical usage of GCM with AES [26]. We refer to this value as

the minimal core-ciphertext length of the scheme $\mathsf{SE1}$, formally defining $\mathsf{SE1.mccl} = \min_{N,M,H} \{\mathsf{SE1.ccl}(|N|,|M|,|H|)\}$ where the minimum is over all $(N,M,H) \in \mathsf{SE1.NS} \times \mathsf{SE1.MS} \times \mathsf{SE1.HS}$. This is relevant because some of our transforms need $\mathsf{SE1.mccl}$ to be non-trivial to provide security.

All transforms here use two keys, meaning the key for the constructed NBE2 scheme $\mathsf{SE2}$ is a pair consisting of a key for a PRF and a key for $\mathsf{SE1}$. An implementation can, starting from a single overlying key, derive these sub-keys and store them, so that neither key size nor computational cost increase. This is well understood and is done as part of OCB, GCM and many other designs.

The ciphertext overhead is the bandwidth cost of the transform. We now discuss how to measure it. In the NBE2 scheme $\mathsf{SE2}$ constructed by any of our transforms from an NBE1 scheme $\mathsf{SE1}$, the ciphertext space is the set of strings of some length, $\mathsf{SE2.CS}(\ell_n,\ell_m,\ell_h) = \{0,1\}^{\mathsf{SE2.cl}(\ell_n,\ell_m,\ell_h)}$. Since NBE1 decryption gets the nonce for free while NBE2 decryption must, effectively, communicate it via the ciphertext, the "fair" definition of the ciphertext overhead of the transform is the maximum, over all possible choices of ℓ_n,ℓ_m,ℓ_h, of

$$\mathsf{SE2.cl}(\ell_n,\ell_m,\ell_h) - \mathsf{SE2.ccl}(\ell_n,\ell_m,\ell_h) - \mathsf{SE1.nl}\ .$$

Another way to put it is that the ciphertext overhead is how much longer ciphertexts are in $\mathsf{SE2}$ than in $\mathbf{TN}[\mathsf{SE1}]$. All our transforms have ciphertext overhead zero, meaning are optimal in terms of bandwidth usage.

6.2 The HN1 transform

The idea of our first transform is that a piece of the core ciphertext may be used as a nonce under which to encrypt the actual nonce. Let $\mathsf{SE1}$ be an NBE1 scheme and F a function family with $\mathsf{F.ol} = \mathsf{SE1.nl}$, so that outputs of $\mathsf{F.Ev}$ can be used to mask nonces for $\mathsf{SE1}$. Assume $\mathsf{SE1.mccl} \geq \mathsf{F.il}$, so that an $\mathsf{F.il}$-bit prefix of a core ciphertext can be used as an input to $\mathsf{F.Ev}$. Invertibility of F is not required, so it can, but need not, be a blockcipher. Our **HN1** transform defines NBE2 scheme $\mathsf{SE}_{\mathrm{HN1}} = \mathbf{HN1}[\mathsf{SE1},\mathsf{F}]$ whose encryption and decryption algorithms are shown in Figure 5. A key (K_{F}, K_1) for $\mathsf{SE}_{\mathrm{HN1}}$ is a pair consisting of a key K_{F} for F and a key K_1 for $\mathsf{SE1}$, so that the key space is $\mathsf{SE}_{\mathrm{HN1}}.\mathsf{KS} = \{0,1\}^{\mathsf{F.kl}} \times \mathsf{SE1.KS}$. The message, header and nonce spaces are unchanged. The parsing $Y \| C_1 \leftarrow C_2$ in the second line of the decryption algorithm $\mathsf{SE}_{\mathrm{HN1}}$ is such that $|Y| = \mathsf{SE1.nl}$. The ciphertext overhead is zero. The computational overhead is one call to $\mathsf{F.Ev}$ for each of encryption or decryption.

Theorem 6.1 below says that if the starting NBE1 scheme $\mathsf{SE1}$ is basic-AE1-secure and F is a PRF then the NBE2 scheme $\mathsf{SE}_{\mathrm{HN1}}$ returned by the transform is basic-AE2-secure. We show authenticity and privacy separately —taking advantage of Theorem 4.1 to obtain joint security— not just for simplicity, but because the bounds and assumptions under which security can be established are different. Authenticity of $\mathsf{SE}_{\mathrm{HN1}}$ reduces tightly to that of $\mathsf{SE1}$ and does not require PRF-security of F, as indicated by Equation (1). PRF-security of F is only required for privacy, where there is also an added term in the bound, as indicated by Equation (2). The proof of the following is in Appendix C.

Theorem 6.1 *Let $\mathsf{SE}_{\mathrm{HN1}} = \mathbf{HN1}[\mathsf{SE1},\mathsf{F}]$ be obtained as above. Then, given adversary $A_2 \in \mathcal{A}^{\mathrm{auth2}}_{\text{u-n}}$ we construct adversary $A_1 \in \mathcal{A}^{\mathrm{auth1}}_{\text{u-n}}$ such that*

$$\mathbf{Adv}^{\mathrm{auth2}}_{\mathsf{SE}_{\mathrm{HN1}}}(A_2) \ \leq\ \mathbf{Adv}^{\mathrm{auth1}}_{\mathsf{SE1}}(A_1)\ . \tag{1}$$

Adversary A_2 preserves the resources of A_1. Also, given adversary $A_2 \in \mathcal{A}^{\mathrm{ae2}}_{\text{u-n}} \cap \mathcal{A}^{\mathrm{ae2}}_{\mathrm{priv}}$, making q_n queries to its New *oracle and q_e queries per user to its* Enc *oracle, we construct adversaries $A_1 \in$*

$\underline{\mathsf{SE}_{\mathrm{HN1}}.\mathsf{Enc}((K_{\mathsf{F}}, K_1), N, M, H)}$	$\underline{\mathsf{SE}_{\mathrm{HN1}}.\mathsf{Dec}((K_{\mathsf{F}}, K_1), C_2, H)}$
$C_1 \leftarrow \mathsf{SE1.Enc}(K_1, N, M, H)$	If $(\lvert C_2\rvert < \mathsf{SE1.nl} + \mathsf{F.il})$ then return $\perp$
$x \leftarrow C_1[1..\mathsf{F.il}]$; $P \leftarrow \mathsf{F.Ev}(K_{\mathsf{F}}, x)$	$Y \Vert C_1 \leftarrow C_2$; $x \leftarrow C_1[1..\mathsf{F.il}]$; $P \leftarrow \mathsf{F.Ev}(K_{\mathsf{F}}, x)$
$Y \leftarrow P \oplus N$; $C_2 \leftarrow Y \Vert C_1$	$N \leftarrow P \oplus Y$; $M \leftarrow \mathsf{SE1.Dec}(K_1, N, C_1, H)$
Return C_2	Return M
$\underline{\mathsf{SE}_{\mathrm{HN2}}.\mathsf{Enc}((K_{\mathsf{E}}, K_1), N, M, H)}$	$\underline{\mathsf{SE}_{\mathrm{HN2}}.\mathsf{Dec}((K_{\mathsf{E}}, K_1), C_2, H)}$
$C_1 \leftarrow \mathsf{SE1.Enc}(K_1, N, M, H)$	If $(\lvert C_2\rvert < \mathsf{E.bl})$ then return $\perp$
$(x, y) \leftarrow \mathsf{Spl.Ev}(\ell, C_1)$	$N \Vert x \leftarrow \mathsf{E.In}(K_{\mathsf{E}}, C_2[1..\mathsf{E.bl}])$
$C_{2,1} \leftarrow \mathsf{E.Ev}(K_{\mathsf{E}}, N \Vert x)$	$y \leftarrow C_2[(\mathsf{E.bl}+1)..\lvert C_2\rvert]$; $C_1 \leftarrow \mathsf{Spl.In}(x, y)$
$C_2 \leftarrow C_{2,1} \Vert y$; Return C_2	$M \leftarrow \mathsf{SE1.Dec}(K_1, N, C_1, H)$; Return M
$\underline{\mathsf{SE}_{\mathrm{HN3}}.\mathsf{Enc}((K_{\mathsf{F}}, K_1), N, M, H)}$	$\underline{\mathsf{SE}_{\mathrm{HN3}}.\mathsf{Dec}((K_{\mathsf{F}}, K_1), C_2, H)}$
$N_1 \leftarrow \mathsf{F.Ev}(K_{\mathsf{F}}, N)$	If $(\lvert C_2\rvert < \mathsf{F.ol})$ then return $\perp$
$C_1 \leftarrow \mathsf{SE1.Enc}(K_1, N_1, M, H)$	$N_1 \Vert C_1 \leftarrow C_2$; $M \leftarrow \mathsf{SE1.Dec}(K_1, N_1, C_1, H)$
$C_2 \leftarrow N_1 \Vert C_1$; Return C_2	Return M

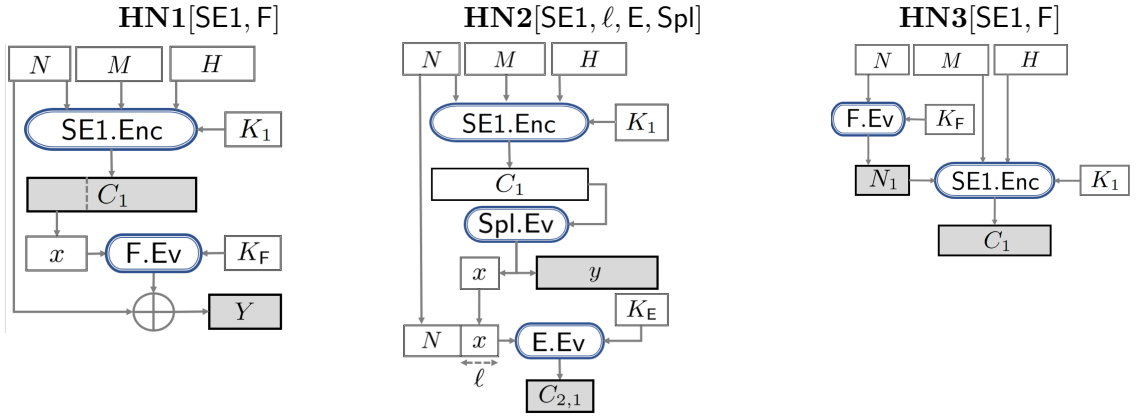

Figure 5: At the top are the encryption and decryption algorithms of the NBE2 schemes constructed by our basic transforms. From top to bottom: $\mathsf{SE}_{\mathrm{HN1}} = \mathbf{HN1}[\mathsf{SE1}, \mathsf{F}]$, $\mathsf{SE}_{\mathrm{HN2}} = \mathbf{HN2}[\mathsf{SE1}, \ell, \mathsf{E}, \mathsf{Spl}]$ and $\mathsf{SE}_{\mathrm{HN3}} = \mathbf{HN3}[\mathsf{SE1}, \mathsf{F}]$. On the bottom are diagrams illustrating the encryption algorithms of the constructed schemes.

$\mathcal{A}^{\mathrm{ae1}}_{\text{u-n}} \cap \mathcal{A}^{\mathrm{ae1}}_{\mathrm{priv}}$ *and* B *such that*

$$\mathbf{Adv}^{\mathrm{ae2}}_{\mathsf{SE}_{\mathrm{HN1}}}(A_2) \;\le\; \mathbf{Adv}^{\mathrm{ae1}}_{\mathsf{SE1}}(A_1) + \mathbf{Adv}^{\mathrm{prf}}_{\mathsf{F}}(B) + \frac{q_n q_e (q_e - 1)}{2^{\mathsf{F.il}+1}} \,. \tag{2}$$

Adversary A_1 *preserves the resources of* A_2*. Adversary* B *makes* q_n *queries to its* NEW *oracle and* q_e *queries per user to its* FN *oracle. Adversary* B *has about the same running time as* A_2*.*

6.3 The HN2 transform

SPLITTING. This transform employs ciphertext stealing [46] to get zero ciphertext overhead. There are many choices with regard to how to implement stealing, for example whether one steals from the first part of the core ciphertext or the last, and implementations may have different preferences.

Accordingly, we do not pin down a choice but instead parameterize the transform by a splitting algorithm responsible for splitting a given string X (the core ciphertext) into segments x (the stolen part, of a prescribed length ℓ) and y (the rest). Formally, splitting scheme Spl specifies a deterministic algorithm $\mathsf{Spl.Ev}$ that takes an integer $\ell \geq 0$ and a string X with $|X| \geq \ell$, and returns a pair of strings $(x, y) \leftarrow \mathsf{Spl.Ev}(\ell, X)$ with $|x| = \ell$. If $(x, y) \in \mathrm{Im}(\mathsf{Spl.Ev}(|x|, \cdot))$ —the image of a function was defined in Section 2— then $X \leftarrow \mathsf{Spl.In}(x, y)$ recovers the unique X such that $\mathsf{Spl.Ev}(|x|, X) = (x, y)$, and otherwise returns $X = \perp$.

This isn't enough because for security we want that if X is random then so are x, y. A simple way to ensure this is to require that the split sets x to some bit positions of X and y to the rest, with the choice of positions depending only on $|X|$. Formally, we require that there is a (deterministic) function $\mathsf{Spl.St}$ that given integers ℓ, n with $n \geq \ell \geq 0$ returns a starting index $s = \mathsf{Spl.St}(\ell, n)$ in the range $1 \leq s \leq n - \ell + 1$, and $\mathsf{Spl.Ev}(\ell, X)$ returns $x = X[s..(s + \ell - 1)]$ and $y = X[1..(s - 1)] \| X[(s + \ell)..|X|]$ for $s = \mathsf{Spl.St}(\ell, |X|)$. The most common choices are that $\mathsf{Spl.St}(\ell, n) = 1$, so that $x = X[1..\ell]$ is the ℓ-bit prefix of X and $y = X[(\ell + 1)..|X|]$ is the rest (corresponding to stealing from the first part of X), or $\mathsf{Spl.St}(\ell, n) = n - \ell + 1$, so that $x = X[(|X| - \ell + 1)..|X|]$ is the ℓ-bit suffix of X and $y = X[1..(|X| - \ell)]$ is the rest (corresponding to stealing from the last part of X), but other choices are possible. Notice that now, assuming it is given inputs of the right lengths, as it will in our usage, $\mathsf{Spl.In}$ will not return $\perp$.

THE **HN2** TRANSFORM. The starting idea of this transform is that our NBE2 scheme can encrypt under the given NBE1 scheme and then also include in the ciphertext an enciphering, under a blockcipher E, of the nonce. We enhance this to encipher, along with the nonce, ℓ bits stolen from the core ciphertext. The stealing has two dividends. First, nonces are often shorter than the block length of E —for example $\mathsf{SE1.nl} = 96$ and $\mathsf{E.bl} = 128$ for AES-GCM and OCB [56, 40]— so in the absence of stealing, the nonce would be padded before enciphering, leading to ciphertext overhead. Second, while we show here (Theorem 6.2) that the scheme preserves basic security regardless of the amount ℓ stolen, we show later (Theorem 7.2) that it preserves even advanced security if ℓ is non-trivial (128 bits or more). We now proceed to the full description.

Let $\mathsf{SE1}$ be an NBE1 scheme, Spl a splitting scheme and $\ell \geq 0$ the prescribed length of the stolen segment of the core ciphertext. We assume the minimal core-ciphertext length of $\mathsf{SE1}$ satisfies $\mathsf{SE1.mccl} \geq \ell$, which ensures that core ciphertexts are long enough to allow the desired splitting. Let E be a blockcipher with block length $\mathsf{E.bl} = \mathsf{SE1.nl} + \ell$. Our **HN2** transform defines NBE2 scheme $\mathsf{SE}_{\mathrm{HN2}} = \mathbf{HN2}[\mathsf{SE1}, \ell, \mathsf{E}, \mathsf{Spl}]$ whose encryption and decryption algorithms are shown in Figure 5. The parsing in the second line of the decryption algorithm $\mathsf{SE}_{\mathrm{HN2}}$ is such that $|N| = \mathsf{SE1.nl}$. A key (K_{E}, K_1) for $\mathsf{SE}_{\mathrm{HN2}}$ is a pair consisting of a key K_{E} for E and a key K_1 for $\mathsf{SE1}$, so that the key space is $\mathsf{SE}_{\mathrm{HN2}}.\mathsf{KS} = \{0,1\}^{\mathsf{E.kl}} \times \mathsf{SE1.KS}$. The nonce, message and header spaces are unchanged. The length of ciphertext C_2 is $\mathsf{E.bl} + |C_1| - \ell = |C_1| + \mathsf{SE1.nl}$, so the ciphertext space is $\mathsf{SE}_{\mathrm{HN2}}.\mathsf{CS}(\ell_n, \ell_m, \ell_h) = \{0,1\}^{\mathsf{SE1.nl}+\mathsf{SE1.ccl}(\ell_n,\ell_m,\ell_h)}$. The ciphertext overhead is zero. The computational overhead is an extra blockcipher call for encryption and a blockcipher inverse for decryption.

A typical instantiation for basic security is $\mathsf{E} = \mathsf{AES}$, so that $\mathsf{E.bl} = 128$. Nonces would have length $\mathsf{SE1.nl} = 96$. We then set $\ell = 32$ and $\mathsf{Spl.St}(\ell, n) = 1$ for all n. This means $\mathsf{SE1.mccl}$ must be at least 32, which is true for all real-world schemes we know. This reduction in the required value of $\mathsf{SE1.mccl}$ for security is a benefit that **HN2** offers over **HN1**. Recall the latter needs $\mathsf{F.il} \geq \mathsf{SE1.mccl}$, and hence by Theorem 6.1 needs $\mathsf{SE1.mccl} \geq 128$, for the same security that **HN2** can offer with $\mathsf{SE1.mccl} \geq 32$.

Theorem 6.2 below says that if the starting NBE1 scheme $\mathsf{SE1}$ is basic-AE1-secure and E is a PRF, then the NBE2 scheme $\mathsf{SE}_{\mathrm{HN2}}$ returned by the transform is basic-AE2-secure. (This holds regardless of the value of ℓ.) We establish authenticity and privacy separately to showcase the

difference in assumptions. Thus authenticity, as per Equation (3) does not require security of the blockcipher E and reduces tightly to the authenticity of $\mathsf{SE1}$. For privacy, which relies on PRF security of E, Equation (4) shows that the reduction is tight, the added term of Equation (2) no longer present. This better bound is another benefit of **HN2** over **HN1**. The proof of the following is in Appendix D.

Theorem 6.2 *Let* $\mathsf{SE}_{\mathrm{HN2}} = \mathbf{HN2}[\mathsf{SE1}, \ell, \mathsf{E}, \mathsf{Spl}]$ *be obtained as above. Then, given adversary* $A_2 \in \mathcal{A}^{\mathrm{auth2}}_{\text{u-n}}$ *we construct adversary* $A_1 \in \mathcal{A}^{\mathrm{auth1}}_{\text{u-n}}$ *such that*

$$\mathbf{Adv}^{\mathrm{auth2}}_{\mathsf{SE}_{\mathrm{HN2}}}(A_2) \;\le\; \mathbf{Adv}^{\mathrm{auth1}}_{\mathsf{SE1}}(A_1)\,. \tag{3}$$

Adversary A_2 *preserves the resources of* A_1*. Also, given adversary* $A_2 \in \mathcal{A}^{\mathrm{ae2}}_{\text{u-n}} \cap \mathcal{A}^{\mathrm{ae2}}_{\mathrm{priv}}$*, making* q_n *queries to its* New *oracle and* q_e *queries per user to its* Enc *oracle, we construct adversaries* $A_1 \in \mathcal{A}^{\mathrm{ae1}}_{\text{u-n}} \cap \mathcal{A}^{\mathrm{ae1}}_{\mathrm{priv}}$ *and* B *such that*

$$\mathbf{Adv}^{\mathrm{ae2}}_{\mathsf{SE}_{\mathrm{HN2}}}(A_2) \;\le\; \mathbf{Adv}^{\mathrm{ae1}}_{\mathsf{SE1}}(A_1) + \mathbf{Adv}^{\mathrm{prf}}_{\mathsf{E}}(B)\,. \tag{4}$$

Adversary A_1 *preserves the resources of* A_2*. Adversary* B *makes* q_n *queries to its* New *oracle and* q_e *queries per user to its* Fn *oracle. Adversary* B *has about the same running time as* A_2*.*

6.4 The HN3 transform

Our third transform uses what we call nonce-based nonce-derivation, in which encryption is performed under $\mathsf{SE1}$ using as nonce the result $N_1 = \mathsf{F}(K_{\mathsf{F}}, N)$ of a PRF F on the actual nonce N. The idea comes from SIV [57] but differences include that: (1) SIV constructs an AE1-secure NBE1 scheme while we construct an AE2-secure NBE2 scheme. (2) SIV decryption needs to have the original nonce. (3) Our synthetic nonce N_1 is a function only of the actual nonce while the one in SIV is also a function of the message and header.

Proceeding to the details, let $\mathsf{SE1}$ be an NBE1 scheme. Let F be a function family with $\mathsf{F.ol} = \mathsf{SE1.nl}$, meaning outputs of $\mathsf{F.Ev}$ can be used as nonces for $\mathsf{SE1}$. Invertibility of F is not required, so it can, but need not, be a blockcipher. Our **HN3** transform defines NBE2 scheme $\mathsf{SE}_{\mathrm{HN3}} = \mathbf{HN3}[\mathsf{SE1}, \mathsf{F}]$ whose encryption and decryption algorithms are shown in Figure 5. A key (K_{F}, K_1) for $\mathsf{SE}_{\mathrm{HN3}}$ is a pair consisting of a key K_{F} for F and a key K_1 for $\mathsf{SE1}$, so that the key space is $\mathsf{SE}_{\mathrm{HN3}}.\mathsf{KS} = \{0,1\}^{\mathsf{F.kl}} \times \mathsf{SE1.KS}$. The message and header spaces are unchanged, and the nonce space is $\mathsf{SE}_{\mathrm{HN3}}.\mathsf{NS} = \{0,1\}^{\mathsf{F.il}}$, meaning inputs to F are nonces for $\mathsf{SE2}$. The parsing in the second line of the decryption algorithm $\mathsf{SE}_{\mathrm{HN3}}$ of Figure 5 is such that $|N_1| = \mathsf{SE1.nl}$. Note that the decryption algorithm does not use F or K_{F}.

As with **HN1** and **HN2**, the **HN3** transform has zero ciphertext overhead. The computational overhead for encryption is one invocation of F. Advantages emerge with decryption, where there is now *no* computational overhead. Indeed decryption in $\mathsf{SE}_{\mathrm{HN3}}$ is effectively the same as in $\mathsf{SE1}$. In particular, in the typical case that F is a blockcipher on which $\mathsf{SE1}$ is itself based, decryption (unlike with **HN2**) no longer needs to implement its inverse, which can be a benefit in hardware and for reducing code size.

The assumed PRF security of F means that the nonce N_1 provided to $\mathsf{SE1.Enc}$ is effectively random. This makes it simple and natural, in proving security, to assume $\mathsf{SE1}$ is AE1$[\mathcal{A}^{\mathrm{ae1}}_{\text{r-n}}]$-secure (recall this is AE1-security for the class of adversaries that pick the nonce at random). Theorem 6.3 below accordingly says that if the starting NBE1 scheme $\mathsf{SE1}$ is AE1$[\mathcal{A}^{\mathrm{ae1}}_{\text{r-n}}]$-secure and F is a PRF then the NBE2 scheme $\mathsf{SE}_{\mathrm{HN1}}$ returned by the transform is basic-AE2-secure. The gap to the assumed basic-AE1-security of $\mathsf{SE1}$ is bridged by applying Theorem 4.3. The proof of the following is in Appendix E.

$\underline{\mathsf{SE}_{\mathrm{HN4}}.\mathsf{Enc}((K_{\mathsf{F}}, K_1), N, M, H)}$	$\underline{\mathsf{SE}_{\mathrm{HN4}}.\mathsf{Dec}((K_{\mathsf{F}}, K_1), C_2, H)}$
$N_1 \leftarrow \mathsf{F.Ev}(K_{\mathsf{F}}, (N, M, H))$	If $(\lvert C_2\rvert < \mathsf{F.ol})$ then return $\perp$
$C_1 \leftarrow \mathsf{SE1.Enc}(K_1, N_1, N\|M, H)$	$N_1\|C_1 \leftarrow C_2$; $X \leftarrow \mathsf{SE1.Dec}(K_1, N_1, C_1, H)$
$C_2 \leftarrow N_1\|C_1$	If $(X = \perp)$ then return $\perp$
Return C_2	$N\|M \leftarrow X$; $T \leftarrow \mathsf{F.Ev}(K_{\mathsf{F}}, (N, M, H))$
	If $(T = N_1)$ then return M else return $\perp$
$\underline{\mathsf{SE}_{\mathrm{HN5}}.\mathsf{Enc}(K_{\mathsf{TE}}, N, M, H)}$	$\underline{\mathsf{SE}_{\mathrm{HN5}}.\mathsf{Dec}(K_{\mathsf{TE}}, C_2, H)}$
$C_2 \leftarrow \mathsf{TE.Ev}(K_{\mathsf{TE}}, H, 0^{\ell_z}\|N\|M)$	$X \leftarrow \mathsf{TE.In}(K_{\mathsf{TE}}, H, C_2)$
Return C_2	If $X[1..\ell_z] \neq 0^{\ell_z}$ then return $\perp$
	$N\|M \leftarrow X[(\ell_z + 1)..\lvert X\rvert]$; Return M

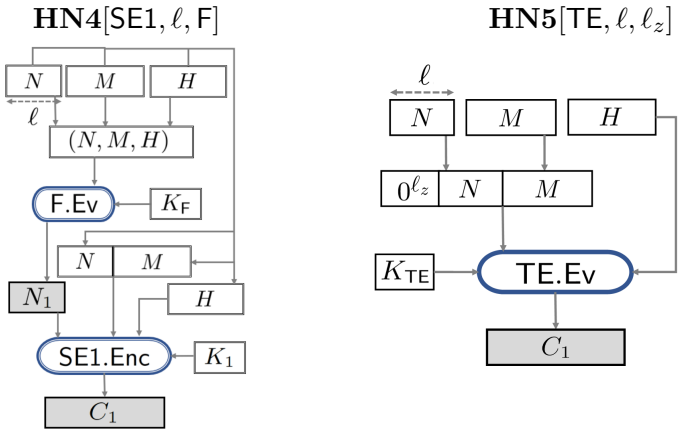

Figure 6: At the top are encryption and decryption algorithms of the NBE2 schemes constructed by our advanced transforms. From top to bottom: $\mathsf{SE}_{\mathrm{HN4}} = \mathbf{HN4}[\mathsf{SE1}, \ell, \mathsf{F}]$ and $\mathsf{SE}_{\mathrm{HN5}} = \mathbf{HN5}[\mathsf{TE}, \ell, \ell_z]$. On the bottom are diagrams illustrating the encryption algorithms of the constructed schemes.

Theorem 6.3 *Let* $\mathsf{SE}_{\mathrm{HN3}} = \mathbf{HN3}[\mathsf{SE1}, \mathsf{F}]$ *be obtained as above. Then, given adversary* $A_2 \in \mathcal{A}^{\mathrm{ae2}}_{\mathrm{u\text{-}n}}$*, making* q_n *queries to its* New *oracle and* q_e *queries per user to its* Enc *oracle, we construct adversaries* $A_1 \in \mathcal{A}^{\mathrm{ae1}}_{\mathrm{r\text{-}n}}$ *and* B *such that*

$$\mathbf{Adv}^{\mathrm{ae2}}_{\mathsf{SE}_{\mathrm{HN3}}}(A_2) \;\leq\; \mathbf{Adv}^{\mathrm{ae1}}_{\mathsf{SE1}}(A_1) + \mathbf{Adv}^{\mathrm{prf}}_{\mathsf{F}}(B)\;. \tag{5}$$

Adversary A_1 *preserves the resources of* A_2*. Adversary* B *makes* q_n *queries to its* New *oracle and* q_e *queries per user to its* Fn *oracle. Adversary* B *has about the same running time as* A_2*.*

7 Advanced transforms

We now turn to achieving AE2-security in the nonce-misuse setting, which we formalized as AE2$[\mathcal{A}^{\mathrm{ae2}}_{\mathrm{u\text{-}nmh}}]$-security. We discuss various transforms for this purpose.

7.1 Advanced security of HN1

We showed in Theorem 6.1 that **HN1** preserves basic security. It turns out that it also preserves advanced security. Theorem 7.1 below says that if the starting NBE1 scheme SE1 is advanced-AE1-

secure and F is a PRF then the NBE2 scheme $\mathsf{SE}_{\mathrm{HN1}}$ returned by the transform is advanced-AE2-secure. The change in the statement compared to Theorem 6.1 is only with regard to the adversary classes changing from unique nonce (basic security) to unique nonce-message-header (advanced security). Again, Equation (6) tightly reduces authenticity of $\mathsf{SE}_{\mathrm{HN1}}$ to that of $\mathsf{SE1}$ and makes no security assumptions on F, while the privacy claim of Equation (7) relies on PRF-security of F. The proof is in Appendix C.

Theorem 7.1 *Let* $\mathsf{SE}_{\mathrm{HN1}} = \mathbf{HN1}[\mathsf{SE1}, \mathsf{F}]$ *be obtained as above. Then, given adversary* $A_2 \in \mathcal{A}^{\mathrm{auth2}}_{\text{u-nmh}}$ *we construct adversary* $A_1 \in \mathcal{A}^{\mathrm{auth1}}_{\text{u-nmh}}$ *such that*

$$\mathbf{Adv}^{\mathrm{auth2}}_{\mathsf{SE}_{\mathrm{HN1}}}(A_2) \;\le\; \mathbf{Adv}^{\mathrm{auth1}}_{\mathsf{SE1}}(A_1)\;. \tag{6}$$

Adversary A_2 *preserves the resources of* A_1*. Also, given adversary* $A_2 \in \mathcal{A}^{\mathrm{ae2}}_{\text{u-nmh}} \cap \mathcal{A}^{\mathrm{ae2}}_{\mathrm{priv}}$*, making* q_n *queries to its* New *oracle and* q_e *queries per user to its* Enc *oracle, we construct adversaries* $A_1 \in \mathcal{A}^{\mathrm{ae1}}_{\text{u-nmh}} \cap \mathcal{A}^{\mathrm{ae1}}_{\mathrm{priv}}$ *and* B *such that*

$$\mathbf{Adv}^{\mathrm{ae2}}_{\mathsf{SE}_{\mathrm{HN1}}}(A_2) \;\le\; \mathbf{Adv}^{\mathrm{ae1}}_{\mathsf{SE1}}(A_1) + \mathbf{Adv}^{\mathrm{prf}}_{\mathsf{F}}(B) + \frac{q_n q_e(q_e-1)}{2^{\mathsf{F.il}+1}}\;. \tag{7}$$

Adversary A_1 *preserves the resources of* A_2*. Adversary* B *makes* q_n *queries to its* New *oracle and* q_e *queries per user to its* Fn *oracle. Adversary* B *has about the same running time as* A_2*.*

7.2 Advanced security of HN2

We showed in Theorem 6.2 that **HN2** preserves basic security regardless of the amount ℓ of stolen core-ciphertext, even $\ell = 0$. For small ℓ, however, **HN2** can leak information about the nonce in the advanced (misuse resistance) setting, so that the resulting scheme does not provide AE2$[\mathcal{A}^{\mathrm{ae2}}_{\text{u-nmh}}]$-security.

To see how **HN2** can reveal information about the nonce, consider the case that $\ell = 0$. Now if two different message-header pairs are encrypted with the same nonce, then the first part of the ciphertext is the same, leading to an $\mathcal{A}^{\mathrm{ae2}}_{\text{u-nmh}}$-adversary with advantage $1 - 2^{-\mathsf{E.bl}}$. The advantage of this attack however decreases (exponentially) as ℓ increases. The following theorem says that once ℓ is non-trivial (say, 128 bits or more), the transform actually preserves advanced security as well. The proof is in Appendix F.

Theorem 7.2 *Let* $\mathsf{SE}_{\mathrm{HN2}} = \mathbf{HN2}[\mathsf{SE1}, \ell, \mathsf{E}, \mathsf{Spl}]$ *be obtained as above. Then, given adversary* $A_2 \in \mathcal{A}^{\mathrm{auth2}}_{\text{u-nmh}}$ *we construct adversary* $A_1 \in \mathcal{A}^{\mathrm{auth1}}_{\text{u-nmh}}$ *such that*

$$\mathbf{Adv}^{\mathrm{auth2}}_{\mathsf{SE}_{\mathrm{HN2}}}(A_2) \;\le\; \mathbf{Adv}^{\mathrm{auth1}}_{\mathsf{SE1}}(A_1)\;. \tag{8}$$

Adversary A_2 *preserves the resources of* A_1*. Also, given adversary* $A_2 \in \mathcal{A}^{\mathrm{ae2}}_{\text{u-nmh}} \cap \mathcal{A}^{\mathrm{ae2}}_{\mathrm{priv}}$*, making* q_n *queries to its* New *oracle and* q_e *queries per user to its* Enc *oracle, we construct adversaries* $A_1 \in \mathcal{A}^{\mathrm{ae1}}_{\text{u-nmh}} \cap \mathcal{A}^{\mathrm{ae1}}_{\mathrm{priv}}$ *and* B *such that*

$$\mathbf{Adv}^{\mathrm{ae2}}_{\mathsf{SE}_{\mathrm{HN2}}}(A_2) \;\le\; \mathbf{Adv}^{\mathrm{ae1}}_{\mathsf{SE1}}(A_1) + \mathbf{Adv}^{\mathrm{prf}}_{\mathsf{E}}(B) + \frac{q_n q_e(q_e-1)}{2^{\ell+1}}\;. \tag{9}$$

Adversary A_1 *preserves the resources of* A_2*. Adversary* B *makes* q_n *queries to its* New *oracle and* q_e *queries per user to its* Fn *oracle. Adversary* B *has about the same running time as* A_2*.*

The above-sketched attack for the $\ell = 0$ case can be extended to an attack (adversary) that for arbitrary ℓ achieves an advantage of about $q_n q_e^2 \cdot 2^{-\ell}$, showing the bound of Theorem 7.2 is essentially tight. The idea is that the adversary can win when the ℓ stolen bits are the same

across two ciphertexts encrypted to the same user. This extends an attack of [58] on Meyer-Matyas ciphertext stealing.

The result of Theorem 7.2, however, is not ideal, because security would need $\ell = 128$, which requires $\mathsf{SE1.mccl} \geq 128$ (not always true) and also, assuming 96-bit nonces, would require that the blockcipher E have block length 128+96=224, which precludes AES. We now give further transforms that do better.

7.3 The HN4 transform

The **HN3** transform clearly does *not* provide advanced-AE2-security because, if a nonce is repeated, the resulting ciphertexts have the same synthetic nonce, and hence the same first parts, which an adversary can notice. The starting idea for **HN4** is to obtain the synthetic nonce N_1 by applying the PRF F, not just to the actual nonce N as in **HN3**, but, as in SIV [57], to (N, M, H). If we now encrypt with N_1 under an NBE1 scheme $\mathsf{SE1}$, we can indeed show that AE2$[\mathcal{A}^{\mathrm{ae2}}_{\text{u-nmh}}]$-security is achieved, assuming $\mathsf{SE1}$ is AE1$[\mathcal{A}^{\mathrm{ae1}}_{\text{u-nmh}}]$-secure. The latter assumption, however, is not satisfactory here because AE1$[\mathcal{A}^{\mathrm{ae1}}_{\text{u-nmh}}]$-security (typically achieved via SIV itself) already requires two passes through the entire input, so our computation of N_1 adds another entire pass, resulting in significant (non-constant) computational overhead. To avoid this we ask whether it would be enough for $\mathsf{SE1}$ to provide only privacy, meaning be AE1$[\mathcal{A}^{\mathrm{ae1}}_{\text{r-n}} \cap \mathcal{A}^{\mathrm{ae1}}_{\text{priv}}]$-secure, because this can be achieved in one pass. Indeed, this is what SIV assumes, but the difficulty is that SIV decryption makes crucial use of the original nonce N to provide authenticity, recomputing it and checking that it matches the one in the ciphertext. But to be nonce hiding, we cannot transmit N. We resolve this by including N as part of the message encrypted under $\mathsf{SE1}$.

Proceeding to the details, let $\mathsf{SE1}$ be an NBE1 scheme. Let F be a function family with $\mathsf{F.ol} = \mathsf{SE1.nl}$, meaning outputs of $\mathsf{F.Ev}$ can be used as nonces for $\mathsf{SE1}$, and also with $\mathsf{SE1.NS} \times \mathsf{SE1.MS} \times \mathsf{SE1.HS} \subseteq \mathsf{F.D}$, meaning triples (N, M, H) can be used as inputs to F. Let $\ell \geq 1$ be an integer prescribing the nonce length of the constructed scheme. Our **HN4** transform defines NBE2 scheme $\mathsf{SE}_{\mathrm{HN4}} = \mathbf{HN4}[\mathsf{SE1}, \ell, \mathsf{F}]$ whose encryption and decryption algorithms are shown in Figure 6. A key (K_{F}, K_1) for $\mathsf{SE}_{\mathrm{HN4}}$ is a pair consisting of a key K_{F} for F and a key K_1 for $\mathsf{SE1}$, so that the key space is $\mathsf{SE}_{\mathrm{HN4}}.\mathsf{KS} = \{0,1\}^{\mathsf{F.kl}} \times \mathsf{SE1.KS}$. The message and header spaces are unchanged, and the nonce space is $\mathsf{SE}_{\mathrm{HN4}}.\mathsf{NS} = \{0,1\}^{\ell}$. The parsing in the second line of the decryption algorithm $\mathsf{SE}_{\mathrm{HN4}}$ of Figure 5 is such that $|N_1| = \mathsf{SE1.nl}$. The ciphertext overhead is zero, and if $\mathsf{SE1}$ is a standard one-pass privacy only scheme like counter-mode, then the computational overhead is constant.

Security, as with SIV, requires that $\mathsf{SE1}$ satisfies tidiness [49]. Formally, for all K, N, C_1, H, if $\mathsf{SE1.Dec}(K, N, C_1, H) = M \neq \perp$ then $\mathsf{SE1.Enc}(K, N, M, H) = C_1$. Our assumption on $\mathsf{SE1}$ is AE1$[\mathcal{A}^{\mathrm{ae1}}_{\text{r-n}} \cap \mathcal{A}^{\mathrm{ae1}}_{\text{priv}}]$-security. (Privacy only, and again, for convenience, for random nonces.) By Theorem 4.3 this is implied by AE1$[\mathcal{A}^{\mathrm{ae1}}_{\text{u-n}} \cap \mathcal{A}^{\mathrm{ae1}}_{\text{priv}}]$-security. Assuming additionally that F is a PRF, the following says that $\mathbf{HN4}[\mathsf{SE1}, \ell, \mathsf{F}]$ is AE2$[\mathcal{A}^{\mathrm{ae2}}_{\text{u-nmh}}]$-secure.

As we have often done before, we consider privacy and authenticity separately to show that the assumptions required, and bounds obtained, differ. Namely, assuming F is a PRF (1) privacy of $\mathsf{SE}_{\mathrm{HN4}} = \mathbf{HN4}[\mathsf{SE1}, \ell, \mathsf{F}]$ is inherited from that of $\mathsf{SE1}$ with a tight reduction and (2) authenticity of $\mathsf{SE}_{\mathrm{HN4}}$ assumes only the tidiness (not privacy) of $\mathsf{SE1}$. The proof is in Appendix G.

Theorem 7.3 *Let* $\mathsf{SE}_{\mathrm{HN4}} = \mathbf{HN4}[\mathsf{SE1}, \ell, \mathsf{F}]$ *be obtained as above, and assume* $\mathsf{SE1}$ *satisfies tidiness. Then, given adversary* $A_2 \in \mathcal{A}^{\mathrm{ae2}}_{\text{u-nmh}} \cap \mathcal{A}^{\mathrm{ae2}}_{\text{priv}}$ *making* q_n *queries to its* NEW *oracle and* q_e *queries per user to its* ENC *oracle, we construct adversaries* $A_1 \in \mathcal{A}^{\mathrm{ae1}}_{\text{r-n}} \cap \mathcal{A}^{\mathrm{ae1}}_{\text{priv}}$ *and* B_1 *such that*

$$\mathbf{Adv}^{\mathrm{ae2}}_{\mathsf{SE2}}(A_2) \leq \mathbf{Adv}^{\mathrm{prf}}_{\mathsf{F}}(B_1) + \mathbf{Adv}^{\mathrm{ae1}}_{\mathsf{SE1}}(A_1)\ . \tag{10}$$

Game $\mathbf{G}^{\mathrm{prf}}_{\mathsf{TE}}$	Game $\mathbf{G}^{\mathrm{prp\text{-}cca}}_{\mathsf{TE}}$
procedure INIT	procedure INIT
$b \leftarrow\!\!\$\ \{0,1\}$	$b \leftarrow\!\!\$\ \{0,1\}$
procedure NEW	procedure NEW
$v \leftarrow v + 1$	$v \leftarrow v + 1$
If $b = 1$ then	If $b = 1$ then
$K_v \leftarrow\!\!\$\ \{0,1\}^{\mathsf{TE.kl}}$	$K_v \leftarrow\!\!\$\ \{0,1\}^{\mathsf{TE.kl}}$
For all $T \in \mathsf{TE.TS}$ do	For all $T \in \mathsf{TE.TS}$ do
$f_{v,T} \leftarrow \mathsf{TE.Ev}(K_v, T, \cdot)$	$\pi_{v,T} \leftarrow \mathsf{TE.Ev}(K_v, T, \cdot)$
Else For all $T \in \mathsf{TE.TS}$ do	Else For all $T \in \mathsf{TE.TS}$ do
$f_{v,T} \leftarrow\!\!\$\ \mathrm{LFUNC}$	$\pi_{v,T} \leftarrow\!\!\$\ \mathrm{LPERM}$
procedure FN(i, T, X)	procedure FN(i, T, X)
Return $f_{i,T}(X)$	Return $\pi_{v,T}(X)$
procedure FIN(b')	procedure FN$^{-1}(i, T, Y)$
Return $(b = b')$	Return $\pi^{-1}_{i,T}(Y)$
	procedure FIN(b')
	Return $(b = b')$

Figure 7: Game defining (multi-user) PRF security for tweakable cipher TE (left) and game defining (multi-user) PRP-CCA security for TE (right).

Adversary A_1 preserves the resources of A_2 up to increasing the lengths of messages in ENC *queries by ℓ. Adversary B_1 makes q_n queries to its* NEW *oracle, and q_e queries to its* FN *oracle per user, and its running time is about that of A_2. Also, given adversary $A_2 \in \mathcal{A}^{\mathrm{auth2}}_{\text{u-nmh}}$ making q_n queries to its* NEW *oracle, q_e queries per user to its* ENC *oracle and q_v queries per user to its* VF *oracle, we construct adversary B_2 such that*

$$\mathbf{Adv}^{\mathrm{auth2}}_{\mathsf{SE2}}(A_2) \leq \mathbf{Adv}^{\mathrm{prf}}_{\mathsf{F}}(B_2) + \frac{q_n q_v}{2^{\mathsf{SE1.nl}}} \ . \tag{11}$$

Adversary B_2 makes q_n queries to its NEW *oracle, and $q_e + q_v$ queries per user to its* FN *oracle, and its running time is about that of A_2.*

7.4 The HN5 transform

Our final transform **HN5** is different. It does not start from an NBE1 scheme but rather from a (arbitrary-input-length) tweakable cipher, extending the encode-then-encipher paradigm [16] to provide advanced-AE2-security. Instantiation via a fast tweakable cipher like AEZ [33] results in correspondingly fast advanced-AE2-secure NBE2.

We encipher the nonce, message and some redundancy, using the header as the tweak. The change from [33] is to move the nonce from tweak to an input so as to hide it, which we will show is enough to confer AE2-security.

TWEAKABLE CIPHERS. These are the basic tool for this transform, so we recall definitions. A tweakable cipher TE [42, 33] specifies a deterministic evaluation algorithm $\mathsf{TE.Ev} : \{0,1\}^{\mathsf{TE.kl}} \times \mathsf{TE.TS} \times \{0,1\}^* \to \{0,1\}^*$ and a deterministic inversion algorithm $\mathsf{TE.In} : \{0,1\}^{\mathsf{TE.kl}} \times \mathsf{TE.TS} \times \{0,1\}^* \to \{0,1\}^*$. Here, $\mathsf{TE.kl}$ is the key length and $\mathsf{TE.TS}$ is the tweak space. We require

that for all $K \in \{0,1\}^{\mathsf{TE.kl}}$, $T \in \mathsf{TE.TS}$ and $X \in \{0,1\}^*$ we have $|\mathsf{TE.Ev}(K,T,X)| = |X|$ and $\mathsf{TE.In}(K,T,\mathsf{TE.Ev}(K,T,X)) = X$.

We define (multi-user) PRF security for tweakable cipher TE via the game $\mathbf{G}^{\mathrm{prf}}_{\mathsf{TE}}(A)$ in Fig. 7. Here LFUNC is the set of all length-preserving functions $f : \{0,1\}^* \to \{0,1\}^*$. It is required that any $\textsc{Fn}(i,T,X)$ query of the adversary A satisfies $i \le v$, $T \in \mathsf{TE.TS}$ and $X \in \{0,1\}^*$. The (multi-user) PRF advantage of A is $\mathbf{Adv}^{\mathrm{prf}}_{\mathsf{TE}}(A) = 2\Pr[\mathbf{G}^{\mathrm{prf}}_{\mathsf{TE}}(A)] - 1$

We define (multi-user) PRP-CCA security [42] for tweakable cipher TE via the game $\mathbf{G}^{\text{prp-cca}}_{\mathsf{TE}}(A)$ in Fig. 7. Here LPERM is the set of all length-preserving bijections $\pi : \{0,1\}^* \to \{0,1\}^*$. (Note that for any such π and any n, restricting π to $\{0,1\}^n$ yields a permutation on $\{0,1\}^n$.) It is required that any $\textsc{Fn}(i,T,X)$ or $\textsc{Fn}^{-1}(i,T,Y)$ query of adversary A satisfies $i \le v$, $T \in \mathsf{TE.TS}$ and $X, Y \in \{0,1\}^*$. The (multi-user) PRP-CCA advantage of A is $\mathbf{Adv}^{\text{prp-cca}}_{\mathsf{TE}}(A) = 2\Pr[\mathbf{G}^{\text{prp-cca}}_{\mathsf{TE}}(A)] - 1$.

The **HN5** transform. Proceeding to the details, let TE be a tweakable cipher as defined in Section 2. Let $\ell \ge 1$ be an integer prescribing the nonce length of the constructed scheme. Let $\ell_z \ge 0$ be the number of bits of redundancy we introduce to provide authenticity [16]. Our transform defines NBE2 scheme $\mathsf{SE}_{\mathrm{HN5}} = \mathbf{HN5}[\mathsf{TE}, \ell, \ell_z]$ whose encryption and decryption algorithms are shown in Figure 6. The key space of $\mathsf{SE}_{\mathrm{HN5}}$ is the key space of TE. The message space is $\{0,1\}^*$. The header space $\mathsf{SE}_{\mathrm{HN5}}.\mathsf{HS}$ is set to the tweak space $\mathsf{TE.TS}$ of TE. The nonce space is $\mathsf{SE}_{\mathrm{HN5}}.\mathsf{NS} = \{0,1\}^\ell$. The length of ciphertext $\mathsf{SE}_{\mathrm{HN5}}.\mathsf{Enc}(K,N,M,H)$ is $\ell_z + |N| + |M|$, so $\mathsf{SE}_{\mathrm{HN5}}.\mathsf{CS}(\ell_n,\ell_m,\ell_h) = \{0,1\}^{\ell_z+\ell+\ell_m}$. Ciphertext overhead, in this case, is not relative to an underlying NBE1 scheme, since there isn't any, but we see that ciphertexts are longer than message plus nonce by just ℓ_z bits, which is effectively optimal [33].

With this transform, it is helpful to establish privacy and authenticity separately because the security notions required to tightly bound them differ. The privacy of $\mathsf{SE}_{\mathrm{HN5}}$ reduces to the PRF security of TE while its authenticity depends on TE being an PRP-CCA secure tweakable cipher and ℓ_z being sufficiently large. The following theorem captures this formally. The proof in is Appendix H.

Theorem 7.4 *Let $\mathsf{SE}_{\mathrm{HN5}} = \mathbf{HN5}[\mathsf{TE}, \ell, \ell_z]$ be obtained as above. Then, given adversary $A_2 \in \mathcal{A}^{\mathrm{ae2}}_{\text{u-nmh}} \cap \mathcal{A}^{\mathrm{ae2}}_{\mathrm{priv}}$, making q_n queries to its* New *oracle and q_e queries per user to its* Enc *oracle, we construct adversary B_1 such that*

$$\mathbf{Adv}^{\mathrm{ae2}}_{\mathsf{SE}_{\mathrm{HN5}}}(A) \;\le\; \mathbf{Adv}^{\mathrm{prf}}_{\mathsf{TE}}(B_1)\,. \tag{12}$$

Adversary B makes q_n queries to its New *oracle and q_e queries per user to its* Fn *oracle, and its running time is about that of A. Also, given adversary $A_2 \in \mathcal{A}^{\mathrm{auth2}}_{\text{u-nmh}}$ making q_n queries to its* New *oracle and q_e, q_v queries per user to its* Enc, Vf *oracles respectively, with $q_e + q_v \le 2^{\ell+\ell_z-1}$, we construct adversary B_2 such that*

$$\mathbf{Adv}^{\mathrm{auth2}}_{\mathsf{SE}_{\mathrm{HN5}}}(A_2) \;\le\; \mathbf{Adv}^{\text{prp-cca}}_{\mathsf{TE}}(B_2) + \frac{2q_nq_d}{2^{\ell_z}}\,. \tag{13}$$

Adversary B_2 makes q_n queries to its New *oracle and q_e, q_v queries per user to its* $\textsc{Fn}, \textsc{Fn}^{-1}$ *oracles respectively, and its running time is about that of A_2.*

8 Dedicated transform for GCM

We have shown that our generic transforms allow us to immunize NBE1 schemes with low overhead. We now present a transform specific to the GCM NBE1 scheme which is used in TLS. Our transform takes advantage of the underlying structure of GCM to further minimize overhead. We also minimize changes to the scheme so that existing hardware and software can easily adapt.

PADDING FUNCTION. Let $\pi : \{0,1\}^n \to \{0,1\}^n$ be a function. (In the scheme it will be $\mathsf{E.Ev}(K,\cdot)$ for a blockcipher E.) We want to run it in counter mode, defining a function $\mathrm{Pad}^{\pi}_{s,t}$ that takes a nonce $N \in \{0,1\}^*$ of length at most n to return a string (the pad) of length t, with t not necessarily a multiple of n. Integer $s \geq 0$ is the starting point. Recall that if i is an integer then as per Section 2, $\langle i \rangle_m$ is the m-bit representation of i mod 2^m. Now we can define:

$\underline{\mathrm{Pad}^{\pi}_{s,t}(N)}$
$L \leftarrow \lfloor t/n \rfloor$; $e \leftarrow t - nL$; $X \leftarrow \varepsilon$
For $i = 0, \ldots, L-1$ do $X \leftarrow X \,\|\, \pi(N \| \langle s+i \rangle_{n-|N|})$
$X \leftarrow X \,\|\, \pi(N \| \langle s+L \rangle_{n-|N|})[1..e]$
Return X

THE **CAU1** TRANSFORM. Following [18], we generalize GCM via a transform **CAU1**. (We add the "1" to indicate that it is an NBE1 scheme.) Let E be a blockcipher. Let H be a function family with $\mathsf{H.D} = \{0,1\}^* \times \{0,1\}^*$ and $\mathsf{H.ol} = \mathsf{H.kl} = \mathsf{E.bl}$. Let $1 \leq \ell < \mathsf{E.bl}$ be an integer indicating the nonce-length. We associate to these the NBE1 scheme $\mathsf{SE1} = \mathbf{CAU1}[\mathsf{E},\mathsf{H},\ell]$ whose encryption and decryption algorithms are shown at the top of Fig. 8. The key K is a key for E, meaning $\mathsf{SE1.KS} = \{0,1\}^{\mathsf{E.kl}}$. The header space is $\mathsf{SE1.HS} = \{0,1\}^*$. The message space $\mathsf{SE1.MS}$ is the set of strings of length at most $\mathsf{E.bl} \cdot (2^{\mathsf{E.bl}-\ell} - 2)$. The nonce space is $\mathsf{SE1.NS} = \{0,1\}^{\ell}$. In the pseudocode of Fig. 8, the parsing $\tau \| C_1^* \leftarrow C_1$ is such that $|\tau| = \mathsf{E.bl}$, and if parsing fails it is understood that the algorithm returns $\perp$.

AES-GCM, as proposed by McGrew and Viega [45] and standardized by NIST [26], is obtained by setting $\mathsf{E} = \mathsf{AES}$ (so $\mathsf{E.bl} = 128$), $\mathsf{H} = \mathsf{GHASH}$ and $\ell = 96$. It is widely used in practice and proven to provide basic AE1-security (i.e. $\mathrm{AE1}[\mathcal{A}^{\mathrm{ae2}}_{\mathrm{u\text{-}n}}]$-security). $\mathsf{SE1}$ has a fixed-length nonce, reflecting the standardized version of GCM, but a variant with variable-length nonces can be obtained by pre-processing the nonce, as discussed in [45, 36].

OUR **CAU2** TRANSFORM. To provide nonce hiding security, we exploit a feature of NBE1 scheme $\mathsf{SE1} = \mathbf{CAU1}[\mathsf{E},\mathsf{H},\ell]$, namely that the nonce can be obtained from the authentication tag τ. In particular, if $\tau \| C_1^* \leftarrow \mathsf{SE1.Enc}(K,N,M,H)$ and $K_{\mathsf{H}} = \mathsf{E.Ev}(K, 0^{\mathsf{E.bl}})$ then the nonce N can be recovered as the first ℓ bits of

$$y = \mathsf{E.In}(K, \tau \oplus \mathsf{H.Ev}(K_{\mathsf{H}}, (C_1^*, H))) \ .$$

Therefore, in our NBE2 variant $\mathsf{SE2} = \mathbf{CAU2}[\mathsf{E},\mathsf{H},\ell]$, we don't explicitly communicate the nonce but rather have the receiver use the tag to compute y as above, rejecting if the last $\mathsf{E.bl} - \ell$ bits of y are not $\langle 1 \rangle_{\mathsf{E.bl}-\ell}$ and otherwise setting N to the first ℓ bits of y. This can be seen as exploiting the "parsimoniousness" of $\mathbf{TN}[\mathsf{SE1}]$ [15]. Unfortunately, merely doing this results in a loss of authenticity because the decryption procedure will succeed for any given ciphertext with probability $2^{-\mathsf{E.bl}+\ell}$, since this is the probability that *some* nonce with suffix $\langle 1 \rangle_{\mathsf{E.bl}-\ell}$ is recovered. This would be unacceptable in GCM since an adversary would be able to forge valid ciphertexts with probability 2^{-32}. So in order to retain security, we add redundancy to the message before encrypting, specifically prepending it with 0^{ℓ}. Decryption will check that the message returned by $\mathsf{SE1.Dec}$ indeed starts with such a string of 0s. We expect that decryption with a "wrong" nonce leads to a ciphertext that lacks the redundancy. A similar technique is used by ADL [4] in their scheme, GCM-RUP, but for a slightly different variant of GCM.

More formally, let $\mathsf{E},\mathsf{H},\ell$ be as for **CAU1** above. Our transform **CAU2** defines an NBE2 scheme $\mathsf{SE2} = \mathbf{CAU2}[\mathsf{E},\mathsf{H},\ell]$ whose encryption and decryption algorithms are shown at the bottom of Fig. 8. The key, header and nonce spaces are the same as for $\mathsf{SE1} = \mathbf{CAU1}[\mathsf{E},\mathsf{H},\ell]$. To allow

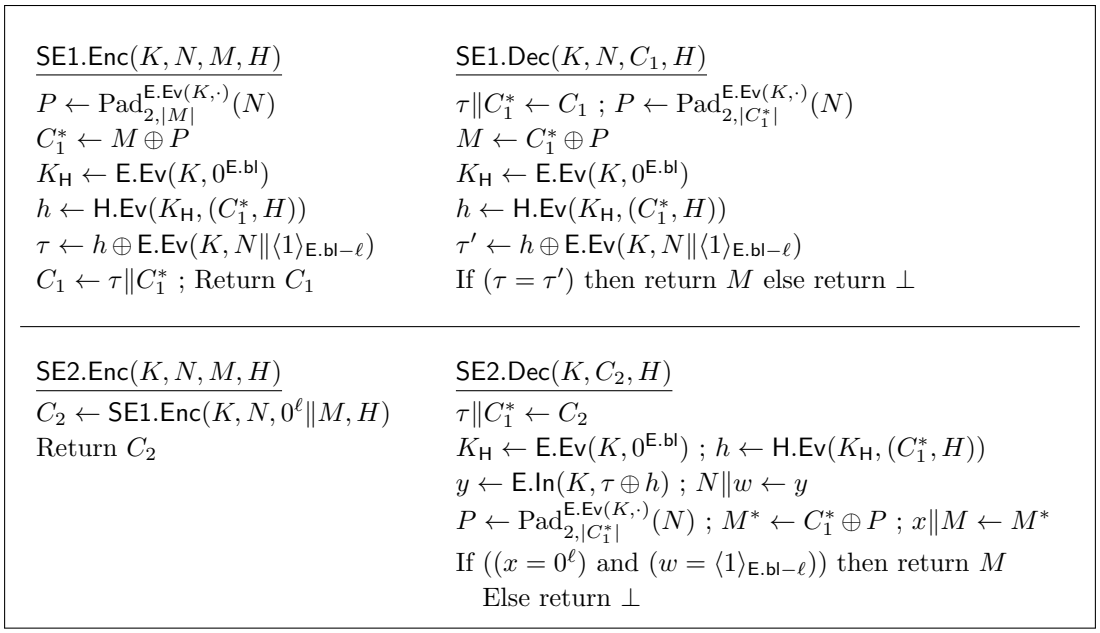

Figure 8: Encryption and decryption algorithms of NBE1 scheme $\mathsf{SE1} = \mathbf{CAU1}[\mathsf{E}, \mathsf{H}, \ell]$ and NBE2 scheme $\mathsf{SE2} = \mathbf{CAU2}[\mathsf{E}, \mathsf{H}, \ell]$. SE2's encryption algorithm uses that of SE1 as a subroutine.

room for the redundancy, the maximum message length is reduced by ℓ bits, so the message space is the set of all strings of length at most $\mathsf{E.bl} \cdot (2^{\mathsf{E.bl}-\ell} - 2) - \ell$. In the pseudocode of Fig. 8, the parsing $N\|w \leftarrow y$ is such that $|N| = \ell$ and $|w| = \mathsf{E.bl} - \ell$. The parsing $x\|M \leftarrow M^*$ is such that $|x| = \ell$, and if parsing fails it is understood that the algorithm returns $\perp$.

Of course an AE2-secure $\mathbf{CAU2}[\mathsf{E}, \mathsf{H}, \ell]$ scheme could be obtained from $\mathbf{CAU1}[\mathsf{E}, \mathsf{H}, \ell]$ via our basic transforms of Section 6, but $\mathbf{CAU2}[\mathsf{E}, \mathsf{H}, \ell]$ has the following advantages over these schemes. It does not change the key, adding no new key materiel. For encryption the code of $\mathbf{CAU1}[\mathsf{E}, \mathsf{H}, \ell]$ can be invoked in a blackbox way, so existing (often extensively optimized) implementations may be reused and existing hardware and software can more easily adapt. Decryption, however, requires more extensive implementation changes.

In the following, we establish basic AE2 security of $\mathbf{CAU2}[\mathsf{E}, \mathsf{H}, \ell]$ assuming PRF-security of E and AXU-security of H. This result improves on the one claimed in the preliminary version of our paper [14], which had needed the stronger assumption that E is a strong PRP. (Meaning, a PRP when the adversary can query both the function and its inverse.) Theorem 4.1 allows us to consider privacy and authenticity separately. As Theorem 8.1 below indicates, privacy is trivially inherited from $\mathbf{CAU1}[\mathsf{E}, \mathsf{H}, \ell]$. The proof for authenticity, namely that of Theorem 8.2, is more invasive and non-trivial.

Privacy of $\mathbf{CAU2}[\mathsf{E}, \mathsf{H}, \ell]$. For privacy of a scheme, only the encryption algorithm is relevant; how decryption is performed makes no difference. Now, as Figure 8 indicates, the encryption algorithm of $\mathsf{SE2} = \mathbf{CAU2}[\mathsf{E}, \mathsf{H}, \ell]$ simply runs that of $\mathsf{SE1} = \mathbf{CAU1}[\mathsf{E}, \mathsf{H}, \ell]$ with 0^ℓ prepended to the message. As a result, privacy of SE2 follows directly from that of SE1:

Theorem 8.1 *Let* $\mathsf{SE1} = \mathbf{CAU1}[\mathsf{E}, \mathsf{H}, \ell]$ *and* $\mathsf{SE2} = \mathbf{CAU2}[\mathsf{E}, \mathsf{H}, \ell]$ *be obtained as above. Then,*

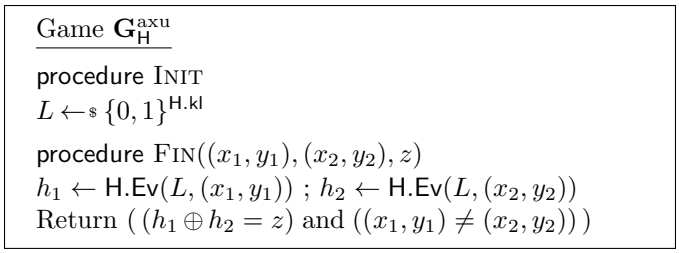

Figure 9: Game defining AXU security for function family H.

---

given adversary $A_2 \in \mathcal{A}^{\text{ae2}}_{\text{priv}} \cap \mathcal{A}^{\text{ae2}}_{\text{u-n}}$ *we construct* $A_1 \in \mathcal{A}^{\text{ae1}}_{\text{priv}} \cap \mathcal{A}^{\text{ae1}}_{\text{u-n}}$ *such that*

$$\mathbf{Adv}^{\text{ae2}}_{\mathsf{SE2}}(A_2) \;\le\; \mathbf{Adv}^{\text{ae1}}_{\mathsf{SE1}}(A_1)\,.$$

Adversary A_1 *preserves the resources of* A_2 *up to an increase of* ℓ *in the lengths of any messages queried to* Enc.

Proof of Theorem 8.1: When A_2 makes a query (i,N,M,H) to its encryption oracle, A_1 queries $(i,N,0^\ell \| M,H)$ to its encryption oracle and returns the result to A_2. Since these are privacy adversaries, there are no decryption queries to consider. When A_2 makes its query to its Fin oracle, adversary A_1 makes the same query to its own Fin oracle. ▌

This allows us to conclude privacy of $\mathsf{SE2} = \mathbf{CAU2}[\mathsf{E},\mathsf{H},\ell]$ based on known proofs and bounds for $\mathsf{SE1} = \mathbf{CAU1}[\mathsf{E},\mathsf{H},\ell]$ from prior work [45, 36, 18, 43, 35]. In particular this allows $\mathsf{SE2}$ to inherit the high-quality bounds shown for $\mathsf{SE1}$ shown by Hoang, Tessaro and Thiruvengadam [35].

AXU security. The authenticity of $\mathsf{SE2} = \mathbf{CAU2}[\mathsf{E},\mathsf{H},\ell]$ assumes axu security of H. We will define a weaker, computational version of the usually information-theoretic definition of [41, 39, 2, 18], and show that this suffices, which makes our results stronger.

Let H be a function family with $\mathsf{H.D} = \{0,1\}^* \times \{0,1\}^*$. Consider game $\mathbf{G}^{\text{axu}}_{\mathsf{H}}$ of Figure 9, and let C be an adversary, that we call an axu-adversary, playing this game. Note that the key L chosen in Init is not returned to the adversary. The adversary has no oracles. To win, it must find, and submit to Fin, a pair $(x_1,y_1),(x_2,y_2)$ of distinct messages, together with the value z of the xor of $\mathsf{H.Ev}(L,\cdot)$ on these messages. We let $\mathbf{Adv}^{\text{axu}}_{\mathsf{H}}(C) = \Pr[\mathbf{G}^{\text{axu}}_{\mathsf{H}}(C)]$ be the probability that the adversary wins.

The advantage of C will depend on the lengths of the inputs in its Fin query. These are accordingly quantified in Theorem 8.2. The computational element of this AXU treatment is that Theorem 8.2 constructs an adversary C with bounded (and specified) resources.

The AXU family GHASH underlying GCM fits in our framework, so our results apply to it. But, unlike prior results, ours apply to other families as well. For example, we could set H to be a PRF or a collision-resistant hash function like SHA256, choices whose security is only computational.

Authenticity of $\mathbf{CAU2}[\mathsf{E},\mathsf{H},\ell]$. We exploit our general results to reduce to as simple a case as possible. (Better bounds may be possible by direct approaches.) First, Theorem 4.2 allows us to restrict attention to a single user. Now, still with a single user, Theorem 4.1 allows us to bound the auth2 advantage for adversaries that are orderly. Finally, a trivial hybrid argument says that, for orderly adversaries, we can assume just one Vf query. Thus, below, the given adversary A_2 against $\mathsf{SE2} = \mathbf{CAU2}[\mathsf{E},\mathsf{H},\ell]$ is assumed to be orderly, to make one New query (single user) and to make one Vf query. The proof, which is the most non-trivial in this paper, is in Appendix I.

Theorem 8.2 *Let* $\mathsf{SE2} = \mathbf{CAU2}[\mathsf{E}, \mathsf{H}, \ell]$ *be obtained as above. Then, given adversary* $A_2 \in \mathcal{A}^{\text{auth2}}_{\text{u-n}} \cap \mathcal{A}^{\text{auth2}}_{\text{ord}}$ *making one query to its* New *oracle,* q_e *queries to its* Enc *oracle and one query to its* Vf *oracle, we construct adversaries* B, C *such that*

$$\mathbf{Adv}^{\text{auth2}}_{\mathsf{SE2}}(A_2) \leq 2 \cdot \mathbf{Adv}^{\text{prf}}_{\mathsf{E}}(B) + q_e \cdot \mathbf{Adv}^{\text{axu}}_{\mathsf{H}}(C) + \frac{1}{2^\ell} \ .$$

Let σ *be the total number of blocks across the messages queried by* A_2 *to* Enc*. Let* m *be the maximum, over all these queries, of the length of the message plus the length of the header in the query. Let* m' *be the length of the ciphertext plus the length of the header in the* Vf *query. Then adversary* B *makes* $\sigma + q_e$ *queries to its* Fn *oracle and its running time is about that of* A_2*. The messages submitted by* C *to* Fin *have lengths at most* $\max(m + \mathsf{E.bl}, m')$ *and the running time of* C *is about that of* A_2*.*

The natural approach to this proof is to begin by switching $\mathsf{E.Ev}(K, \cdot)$ to a random permutation. This would need us to assume prp-cca (strong prp) security because the inverse function is computed in Vf. Instead our proof delays the switch, staying with $\mathsf{E.Ev}(K, \cdot)$ and exploiting its being a permutation to move to a game in which Vf does not need to compute the inverse $\mathsf{E.In}(K, \cdot)$. Once this is done, we can switch $\mathsf{E.Ev}(K, \cdot)$ to a random *function* and rely only on the PRF assumption. Then, another game sequences is used to reduce to the assumed axu-security of H.

A bound on the auth2-advantage of an adversary that makes multiple New and Vf queries can be obtained, as noted above, by combining our general results with Theorem 8.2. An interesting open question is to directly analyze such an adversary and obtain a bound better than ours on its auth2-advantage.

9 A real-world perspective

In addition to bridging the gap between theory and usage, our framework allows us to formalize weaknesses of real-world schemes which communicate nonces in the clear.

First, it allows us to formalize an intuitive fact: pathologically chosen nonces cannot be communicated in the clear. It may seem obvious that message or key-dependent nonces violate security but such pathological nonce choices have occurred in the wild. For instance, CakePHP, a web framework, used the key as the nonce [1] when encrypting data. The use of a hash of a message has also been proposed, and subsequently argued as insecure, in an Internet forum [51].

Second, it disallows metadata leakage through the nonce. Implicit nonces with a device specific field, such as those recommended in RFC 5116 [44] enable an adversary to distinguish between different user sessions. Even the "standard" nonce choices are not safe against these adversaries. A counter will allow an adversary distinguish between sessions with high traffic and low traffic, and a randomly chosen nonce can detect devices with poor entropy (RSA public keys were used to a similar end by HDWH [32]).

10 Acknowledgements

We thank the anonymous reviewers (of the many conferences to which this paper was submitted before finally being accepted at Crypto 2019) for their feedback and suggestions.

References

[1] CakePHP: Using the IV as the key. `http://www.cryptofails.com/post/70059594911/cakephp-using-the-iv-as-the-key`. Accessed: 2019-02-12. 29

[2] F. Abed, S. R. Fluhrer, C. Forler, E. List, S. Lucks, D. A. McGrew, and J. Wenzel. Pipelineable on-line encryption. In C. Cid and C. Rechberger, editors, *FSE 2014*, volume 8540 of *LNCS*, pages 205–223. Springer, Heidelberg, Mar. 2015. 28

[3] E. Andreeva, A. Bogdanov, A. Luykx, B. Mennink, N. Mouha, and K. Yasuda. How to securely release unverified plaintext in authenticated encryption. In P. Sarkar and T. Iwata, editors, *ASIACRYPT 2014, Part I*, volume 8873 of *LNCS*, pages 105–125. Springer, Heidelberg, Dec. 2014. 7

[4] T. Ashur, O. Dunkelman, and A. Luykx. Boosting authenticated encryption robustness with minimal modifications. In J. Katz and H. Shacham, editors, *CRYPTO 2017, Part III*, volume 10403 of *LNCS*, pages 3–33. Springer, Heidelberg, Aug. 2017. 7, 26

[5] J. Aumasson, S. Babbage, D. Bernstein, C. Cid, J. Daemen, O. Dunkelman, K. Gaj, S. Gueron, P. Junod, A. Langley, D. McGrew, K. Paterson, B. Preneel, C. Rechberger, V. Rijmen, M. Robshaw, P. Sarkar, P. Schaumont, A. Shamir, and I. Verbauwhede. CHAE: Challenges in authenticated encryption. ECRYPT-CSA D1.1, Revision 1.05, March 2017. `https://chae.cr.yp.to/whitepaper.html`. 3

[6] M. Barbosa and P. Farshim. Indifferentiable authenticated encryption. In H. Shacham and A. Boldyreva, editors, *CRYPTO 2018, Part I*, volume 10991 of *LNCS*, pages 187–220. Springer, Heidelberg, Aug. 2018. 7

[7] G. Barwell, D. P. Martin, E. Oswald, and M. Stam. Authenticated encryption in the face of protocol and side channel leakage. In T. Takagi and T. Peyrin, editors, *ASIACRYPT 2017, Part I*, volume 10624 of *LNCS*, pages 693–723. Springer, Heidelberg, Dec. 2017. 7

[8] G. Barwell, D. Page, and M. Stam. Rogue decryption failures: Reconciling AE robustness notions. In J. Groth, editor, *15th IMA International Conference on Cryptography and Coding*, volume 9496 of *LNCS*, pages 94–111. Springer, Heidelberg, Dec. 2015. 7

[9] M. Bellare, A. Boldyreva, and S. Micali. Public-key encryption in a multi-user setting: Security proofs and improvements. In B. Preneel, editor, *EUROCRYPT 2000*, volume 1807 of *LNCS*, pages 259–274. Springer, Heidelberg, May 2000. 8

[10] M. Bellare, R. Canetti, and H. Krawczyk. Pseudorandom functions revisited: The cascade construction and its concrete security. In *37th FOCS*, pages 514–523. IEEE Computer Society Press, Oct. 1996. 9

[11] M. Bellare, A. Desai, E. Jokipii, and P. Rogaway. A concrete security treatment of symmetric encryption. In *38th FOCS*, pages 394–403. IEEE Computer Society Press, Oct. 1997. 2

[12] M. Bellare and S. Keelveedhi. Authenticated and misuse-resistant encryption of key-dependent data. In P. Rogaway, editor, *CRYPTO 2011*, volume 6841 of *LNCS*, pages 610–629. Springer, Heidelberg, Aug. 2011. 7

[13] M. Bellare and C. Namprempre. Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. In T. Okamoto, editor, *ASIACRYPT 2000*, volume 1976 of *LNCS*, pages 531–545. Springer, Heidelberg, Dec. 2000. 2, 11, 13

[14] M. Bellare, R. Ng, and B. Tackmann. Nonces are noticed: AEAD revisited. In A. Boldyreva and D. Micciancio, editors, *CRYPTO 2019, Part I*, volume 11692 of *LNCS*, pages 235–265. Springer, Heidelberg, Aug. 2019. 6, 27

[15] M. Bellare and P. Rogaway. On the construction of variable-input-length ciphers. In L. R. Knudsen, editor, *FSE'99*, volume 1636 of *LNCS*, pages 231–244. Springer, Heidelberg, Mar. 1999. 26

[16] M. Bellare and P. Rogaway. Encode-then-encipher encryption: How to exploit nonces or redundancy in plaintexts for efficient cryptography. In T. Okamoto, editor, *ASIACRYPT 2000*, volume 1976 of *LNCS*, pages 317–330. Springer, Heidelberg, Dec. 2000. 2, 6, 24, 25

[17] M. Bellare and P. Rogaway. The security of triple encryption and a framework for code-based game-playing proofs. In S. Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 409–426. Springer, Heidelberg, May / June 2006. 8, 35, 38, 42, 46

[18] M. Bellare and B. Tackmann. The multi-user security of authenticated encryption: AES-GCM in TLS 1.3. In M. Robshaw and J. Katz, editors, *CRYPTO 2016, Part I*, volume 9814 of *LNCS*, pages 247–276. Springer, Heidelberg, Aug. 2016. 7, 11, 26, 28

[19] D. Bernstein. CAESAR call for submissions, final (2014.01.27), 2014. 2, 5

[20] D. J. Bernstein. Re: secret message numbers. Message in Google group on cryptographic competitions, October 2013. `https://groups.google.com/d/msg/crypto-competitions/n5ECGwYr6Vk/bsEfPWqSAU4J`. 3, 7

[21] F. Berti, C. Guo, O. Pereira, T. Peters, and F.-X. Standaert. Tedt, a leakage-resilient aead mode for high (physical) security applications. Cryptology ePrint Archive, Report 2019/137, 2019. `https://eprint.iacr.org/2019/137`. 7

[22] P. Bose, V. T. Hoang, and S. Tessaro. Revisiting AES-GCM-SIV: Multi-user security, faster key derivation, and better bounds. In J. B. Nielsen and V. Rijmen, editors, *EUROCRYPT 2018, Part I*, volume 10820 of *LNCS*, pages 468–499. Springer, Heidelberg, Apr. / May 2018. 4, 5, 6, 7, 11, 13, 14

[23] CAESAR Committee. Cryptographic competitions: Caesar call for submissions, final (2014.01.27). `https://competitions.cr.yp.to/caesar-call.html`. Accessed: 2018-07-23. 7

[24] A. Connolly, P. Farshim, and G. Fuchsbauer. Security of symmetric primitives against key-correlated attacks. *IACR Transactions on Symmetric Cryptology*, pages 193–230, 2019. 7

[25] Y. Dodis, P. Grubbs, T. Ristenpart, and J. Woodage. Fast message franking: From invisible salamanders to encryptment. In H. Shacham and A. Boldyreva, editors, *CRYPTO 2018, Part I*, volume 10991 of *LNCS*, pages 155–186. Springer, Heidelberg, Aug. 2018. 7

[26] M. Dworkin. Recommendation for block cipher modes of operation: Galois/Counter Mode (GCM) and GMAC. NIST Special Publication 800-38D, November 2007. 2, 6, 16, 26

[27] P. Farshim, C. Orlandi, and R. Roşie. Security of symmetric primitives under incorrect usage of keys. *IACR Trans. Symm. Cryptol.*, 2017(1):449–473, 2017. 7

[28] E. Fleischmann, C. Forler, and S. Lucks. McOE: A family of almost foolproof on-line authenticated encryption schemes. In A. Canteaut, editor, *FSE 2012*, volume 7549 of *LNCS*, pages 196–215. Springer, Heidelberg, Mar. 2012. 7

[29] P. Grubbs, J. Lu, and T. Ristenpart. Message franking via committing authenticated encryption. In J. Katz and H. Shacham, editors, *CRYPTO 2017, Part III*, volume 10403 of *LNCS*, pages 66–97. Springer, Heidelberg, Aug. 2017. 7, 12

[30] S. Gueron, A. Langley, and Y. Lindell. AES-GCM-SIV: Specification and analysis. Cryptology ePrint Archive, Report 2017/168, 2017. `http://eprint.iacr.org/2017/168`. 5, 6, 7

[31] S. Gueron and Y. Lindell. GCM-SIV: Full nonce misuse-resistant authenticated encryption at under one cycle per byte. In I. Ray, N. Li, and C. Kruegel, editors, *ACM CCS 2015*, pages 109–119. ACM Press, Oct. 2015. 5, 6

[32] N. Heninger, Z. Durumeric, E. Wustrow, and J. A. Halderman. Mining your ps and qs: Detection of widespread weak keys in network devices. In *USENIX Security Symposium*, volume 8, page 1, 2012. 29

[33] V. T. Hoang, T. Krovetz, and P. Rogaway. Robust authenticated-encryption AEZ and the problem that it solves. In E. Oswald and M. Fischlin, editors, *EUROCRYPT 2015, Part I*, volume 9056 of *LNCS*, pages 15–44. Springer, Heidelberg, Apr. 2015. 6, 24, 25

[34] V. T. Hoang, R. Reyhanitabar, P. Rogaway, and D. Vizár. Online authenticated-encryption and its nonce-reuse misuse-resistance. In R. Gennaro and M. J. B. Robshaw, editors, *CRYPTO 2015, Part I*, volume 9215 of *LNCS*, pages 493–517. Springer, Heidelberg, Aug. 2015. 7

[35] V. T. Hoang, S. Tessaro, and A. Thiruvengadam. The multi-user security of GCM, revisited: Tight bounds for nonce randomization. In D. Lie, M. Mannan, M. Backes, and X. Wang, editors, *ACM CCS 2018*, pages 1429–1440. ACM Press, Oct. 2018. 5, 7, 28

[36] T. Iwata, K. Ohashi, and K. Minematsu. Breaking and repairing GCM security proofs. In R. Safavi-Naini and R. Canetti, editors, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 31–49. Springer, Heidelberg, Aug. 2012. 5, 7, 26, 28

[37] A. Joux. Authentication failures in NIST version of GCM, 2006. Comments submitted to NIST modes of operation process, `https://csrc.nist.gov/csrc/media/projects/block-cipher-techniques/documents/bcm/comments/800-38-series-drafts/gcm/joux_comments.pdf`. 6

[38] J. Katz and M. Yung. Unforgeable encryption and chosen ciphertext secure modes of operation. In B. Schneier, editor, *FSE 2000*, volume 1978 of *LNCS*, pages 284–299. Springer, Heidelberg, Apr. 2001. 2

[39] H. Krawczyk. LFSR-based hashing and authentication. In Y. Desmedt, editor, *CRYPTO'94*, volume 839 of *LNCS*, pages 129–139. Springer, Heidelberg, Aug. 1994. 28

[40] T. Krovetz and P. Rogaway. The software performance of authenticated-encryption modes. In A. Joux, editor, *FSE 2011*, volume 6733 of *LNCS*, pages 306–327. Springer, Heidelberg, Feb. 2011. 2, 5, 19

[41] K. Kurosawa and T. Iwata. TMAC: Two-key CBC MAC. In M. Joye, editor, *CT-RSA 2003*, volume 2612 of *LNCS*, pages 33–49. Springer, Heidelberg, Apr. 2003. 28

[42] M. Liskov, R. L. Rivest, and D. Wagner. Tweakable block ciphers. *Journal of Cryptology*, 24(3):588–613, July 2011. 24, 25

[43] A. Luykx, B. Mennink, and K. G. Paterson. Analyzing multi-key security degradation. In T. Takagi and T. Peyrin, editors, *ASIACRYPT 2017, Part II*, volume 10625 of *LNCS*, pages 575–605. Springer, Heidelberg, Dec. 2017. 7, 28

[44] D. McGrew. An interface and algorithms for authenticated encryption. IETF Network Working Group, RFC 5116, January 2008. 2, 3, 29

[45] D. A. McGrew and J. Viega. The security and performance of the Galois/counter mode (GCM) of operation. In A. Canteaut and K. Viswanathan, editors, *INDOCRYPT 2004*, volume 3348 of *LNCS*, pages 343–355. Springer, Heidelberg, Dec. 2004. 2, 5, 6, 7, 26, 28

[46] C. H. Meyer and S. M. Matyas. *CRYPTOGRAPHY: A new dimension in computer data security: A guide for the design and implementation of secure systems*. Wiley, 1982. 18

[47] K. Minematsu. Authenticated encryption with small stretch (or, how to accelerate AERO). In J. K. Liu and R. Steinfeld, editors, *ACISP 16, Part II*, volume 9723 of *LNCS*, pages 347–362. Springer, Heidelberg, July 2016. 7

[48] C. Namprempre, P. Rogaway, and T. Shrimpton. AE5 security notions: Definitions implicit in the CAESAR call. Cryptology ePrint Archive, Report 2013/242, 2013. `http://eprint.iacr.org/2013/242`. 7

[49] C. Namprempre, P. Rogaway, and T. Shrimpton. Reconsidering generic composition. In P. Q. Nguyen and E. Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 257–274. Springer, Heidelberg, May 2014. 5, 6, 23

[50] T. Peyrin and Y. Seurin. Counter-in-tweak: Authenticated encryption modes for tweakable block ciphers. In M. Robshaw and J. Katz, editors, *CRYPTO 2016, Part I*, volume 9814 of *LNCS*, pages 33–63. Springer, Heidelberg, Aug. 2016. 5

[51] Reddit. Hash of message as nonce?, 2015. `https://redd.it/3c504m`. 29

[52] P. Rogaway. Authenticated-encryption with associated-data. In V. Atluri, editor, *ACM CCS 2002*, pages 98–107. ACM Press, Nov. 2002. 2, 3, 9, 11, 13

[53] P. Rogaway. Efficient instantiations of tweakable blockciphers and refinements to modes OCB and PMAC. In P. J. Lee, editor, *ASIACRYPT 2004*, volume 3329 of *LNCS*, pages 16–31. Springer, Heidelberg, Dec. 2004. 2

[54] P. Rogaway. Nonce-based symmetric encryption. In B. K. Roy and W. Meier, editors, *FSE 2004*, volume 3017 of *LNCS*, pages 348–359. Springer, Heidelberg, Feb. 2004. 2, 9, 11

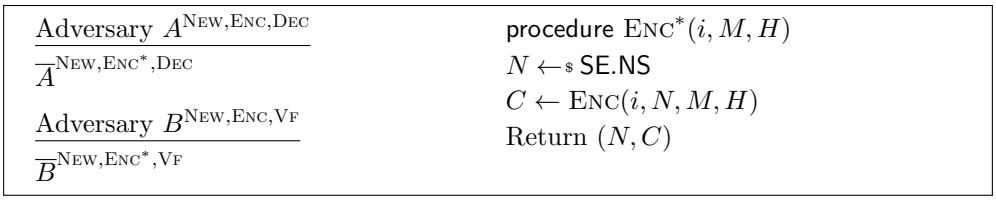

Figure 10: Formalizing random-nonce adversaries.

[55] P. Rogaway. The evolution of authenticated encryption. Real World Cryptography Workshop, Stanford, January 2013. `https://crypto.stanford.edu/RealWorldCrypto/slides/phil.pdf`. 3

[56] P. Rogaway, M. Bellare, J. Black, and T. Krovetz. OCB: A block-cipher mode of operation for efficient authenticated encryption. In M. K. Reiter and P. Samarati, editors, *ACM CCS 2001*, pages 196–205. ACM Press, Nov. 2001. 2, 5, 9, 16, 19

[57] P. Rogaway and T. Shrimpton. A provable-security treatment of the key-wrap problem. In S. Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 373–390. Springer, Heidelberg, May / June 2006. 3, 6, 11, 20, 23

[58] P. Rogaway, M. Wooding, and H. Zhang. The security of ciphertext stealing. In A. Canteaut, editor, *FSE 2012*, volume 7549 of *LNCS*, pages 180–195. Springer, Heidelberg, Mar. 2012. 23

[59] S. Vaudenay and D. Vizár. Under pressure: Security of caesar candidates beyond their guarantees. Cryptology ePrint Archive, Report 2017/1147, 2017. `https://eprint.iacr.org/2017/1147`. 6

[60] H. Wu and B. Preneel. AEGIS: A fast authenticated encryption algorithm. In T. Lange, K. Lauter, and P. Lisonek, editors, *SAC 2013*, volume 8282 of *LNCS*, pages 185–201. Springer, Heidelberg, Aug. 2014. 5

A Adversary classes $\mathcal{A}^{\text{aeX}}_{\text{r-n}}, \mathcal{A}^{\text{authX}}_{\text{r-n}}$

In Section 3, we called an adversary A a random-nonce adversary if the nonces in its queries to Enc are picked uniformly at random, from the (assumed finite) nonce space of the underlying scheme, and independently of each other and anything else. (No restriction is placed on the nonces that the adversary submits in Dec queries.) We had let $\mathcal{A}^{\text{aeX}}_{\text{r-n}}$, where $X \in \{1, 2\}$, be the class of such adversaries for AEX.

When a particular adversary is specified (in pseudocode or otherwise) it is usually easy to tell whether it is random-nonce, but the definition itself still remains somewhat informal. Here we discuss how to bridge this gap.

Let A be an adversary attacking scheme SE, where the latter is an NBEX scheme. Then $A \in \mathcal{A}^{\text{aeX}}_{\text{r-n}}$ if there is another adversary $\overline{A}$, called the core adversary, such that A is defined in terms of $\overline{A}$ as shown in Figure 10. Random-nonce adversaries $B \in \mathcal{A}^{\text{authX}}_{\text{r-n}}$ can be analogously defined, and the core adversaries $\overline{B}$ are shown in the same Figure.

As the Figure indicates, the core adversaries $\overline{A}, \overline{B}$ are given Enc^*, which unlike Enc in $\mathbf{G}^{\text{aeX}}_{\mathsf{SE}}$ or $\mathbf{G}^{\text{authX}}_{\mathsf{SE}}$, takes only i, M, H (no nonce) and returns both a nonce and a ciphertext. (The latter means the random nonces are not "hidden" from $\overline{A}, \overline{B}$.) They access all other oracles in the same way as A, B.

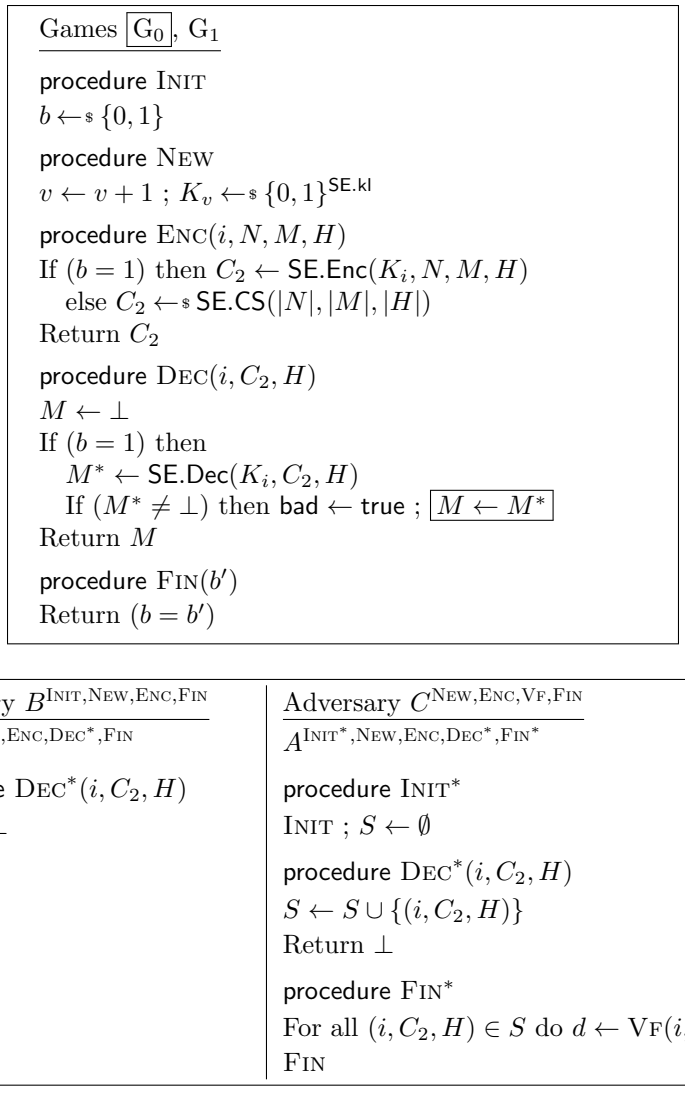

Figure 11: At the top are the games used in proving Theorem 4.1. On the bottom are the adversaries used in proving Theorem 4.1.

B Proof of Theorem 4.1

Proof of Theorem 4.1: We give the proof for X=2, meaning for AE2. The proof for AE1 is analogous.

We assume that A makes no trivial queries. So it does not query $\text{Dec}(i,C_2,H)$ if $\mathbf{M}[i,C_2,H]$ is already defined. In the y=n case, it does not repeat a nonce-user pair in an Enc query, and in the y=nmh case, it does not repeat an Enc query. Games $\mathrm{G}_0, \mathrm{G}_1$ in Fig. 11 are identical-until-bad so using the Fundamental Lemma of Game Playing [17] we have

$$\mathbf{Adv}^{\text{aeX}}_{\mathsf{SE}}(A) = 2\Pr[\mathrm{G}_0(A)] - 1$$

Adversary $A_1^{\text{New},\text{Enc},\text{Vf},\text{Fin}}$

$A_2^{\text{New}^*,\text{Enc}^*,\text{Vf}^*,\text{Fin}}$

procedure New^*
$v \leftarrow v+1$; $K_{\mathsf{F},v} \leftarrow\!\!\$\ \{0,1\}^{\mathsf{F.kl}}$; New

procedure $\text{Enc}^*(i,N,M,H)$
$C_1 \leftarrow \text{Enc}(i,N,M,H)$; $x \leftarrow C_1[1..\mathsf{F.il}]$; $Y \leftarrow N \oplus \mathsf{F.Ev}(K_{\mathsf{F},i},x)$; $C_2 \leftarrow Y \| C_1$
Return C_2

procedure $\text{Vf}^*(i,C_2,H)$
If $(|C_2| < \mathsf{SE1.nl} + \mathsf{F.il})$ then return $\perp$
$Y \| C_1 \leftarrow C_2$; $x \leftarrow C_1[1..\mathsf{F.il}]$; $N \leftarrow Y \oplus \mathsf{F.Ev}(K_{\mathsf{F},i},x)$; Return $\text{Vf}(i,N,C_1,H)$

Figure 12: Adversary A_1 used in proving Equation (14).

$$= 2\Pr[\mathrm{G}_1(A)] - 1 + 2(\Pr[\mathrm{G}_0(A)] - \Pr[\mathrm{G}_1(A)])$$
$$\leq 2\Pr[\mathrm{G}_1(A)] - 1 + 2\Pr[\mathrm{G}_1(A) \text{ sets } \mathsf{bad}] \ .$$

In Fig. 11, we specify adversary B such that

$$2\Pr[\mathrm{G}_1(A)] - 1 \leq \mathbf{Adv}^{\mathrm{aeX}}_{\mathsf{SE}}(B) \ .$$

Adversary B, being a privacy adversary, makes no Dec queries, so we omit this oracle from the list in its superscript. It simulates all queries of A directly, except for additionally returning $\perp$ in response to any Dec query made by A.

In game G_1, flag bad can only be set if $b = 1$, so

$$\Pr[\mathrm{G}_1(A) \text{ sets } \mathsf{bad}] = \frac{1}{2} \cdot \Pr[\,\mathrm{G}_1(A) \text{ sets } \mathsf{bad} \,|\, b = 0\,] + \frac{1}{2} \cdot \Pr[\,\mathrm{G}_1(A) \text{ sets } \mathsf{bad} \,|\, b = 1\,]$$
$$= \frac{1}{2} \cdot \Pr[\,\mathrm{G}_1(A) \text{ sets } \mathsf{bad} \,|\, b = 1\,] \ .$$

In Fig. 11, we specify adversary C such that

$$\Pr[\,\mathrm{G}_1(A) \text{ sets } \mathsf{bad} \,|\, b = 1\,] \leq \mathbf{Adv}^{\mathrm{authX}}_{\mathsf{SE}}(C) \ .$$

Putting all this together concludes the proof. ∎

C Proofs of Theorems 6.1 and 7.1

We prove the following which implies both theorems, Theorem 6.1 being the case y = n and Theorem 7.1 being the case y = nmh. The techniques here are standard and explanations are accordingly kept brief.

Lemma C.1 *Let* $\mathsf{SE}_{\mathrm{HN1}} = \mathbf{HN1}[\mathsf{SE1},\mathsf{F}]$ *be obtained as in Section 6. Let* $\mathrm{y} \in \{\mathrm{n},\mathrm{unm}\}$. *Then, given adversary* $A_2 \in \mathcal{A}^{\mathrm{auth2}}_{\text{u-y}}$ *we construct adversary* $A_1 \in \mathcal{A}^{\mathrm{auth1}}_{\text{u-y}}$ *such that*

$$\mathbf{Adv}^{\mathrm{auth2}}_{\mathsf{SE}_{\mathrm{HN1}}}(A_2) \ \leq \ \mathbf{Adv}^{\mathrm{auth1}}_{\mathsf{SE1}}(A_1) \ . \tag{14}$$

Adversary A_2 *preserves the resources of* A_1. *Also, given adversary* $A_2 \in \mathcal{A}^{\mathrm{ae2}}_{\text{u-y}} \cap \mathcal{A}^{\mathrm{ae2}}_{\mathrm{priv}}$, *making* q_n *queries to its* New *oracle and* q_e *queries per user to its* Enc *oracle, we construct adversaries* $A_1 \in$

procedure $\textsc{Fin}(b')$ // For all games
Return $(b' = 1)$

Games G_0, G_1

procedure $\textsc{New}$
$v \leftarrow v + 1$; $K_{1,v} \leftarrow\!\!\$\ \mathsf{SE1.KS}$
$K_{\mathsf{F},v} \leftarrow\!\!\$\ \{0,1\}^{\mathsf{F.kl}}$; $f_v \leftarrow \mathsf{F.Ev}(K_{\mathsf{F},v}, \cdot)$ // Game G_0
$f_v \leftarrow\!\!\$\ \mathrm{FUNC}(\{0,1\}^{\mathsf{F.il}}, \{0,1\}^{\mathsf{F.ol}})$ // Game G_1

procedure $\textsc{Enc}(i, N, M, H)$
$C_1 \leftarrow \mathsf{SE1.Enc}(K_{1,i}, N, M, H)$
$x \leftarrow C_1[1..\mathsf{F.il}]$; $P \leftarrow f_i(x)$; $Y \leftarrow P \oplus N$
$C_2 \leftarrow Y \| C_1$; Return C_2

Games G_2, $\boxed{\mathrm{G}_3}$

procedure $\textsc{New}$
$v \leftarrow v + 1$
$f_v \leftarrow\!\!\$\ \mathrm{FUNC}(\{0,1\}^{\mathsf{F.il}}, \{0,1\}^{\mathsf{F.ol}})$

procedure $\textsc{Enc}(i, N, M, H)$
$C_1 \leftarrow\!\!\$\ \{0,1\}^{\mathsf{SE1.ccl}(|N|,|M|,|H|)}$
$x \leftarrow C_1[1..\mathsf{F.il}]$; $P \leftarrow f_i(x)$; $Y \leftarrow P \oplus N$
If $(x \in S_i)$ then $\mathsf{bad} \leftarrow \mathsf{true}$; $\boxed{Y \leftarrow\!\!\$\ \{0,1\}^{\mathsf{F.ol}}}$
$S_i \leftarrow S_i \cup \{x\}$
$C_2 \leftarrow Y \| C_1$; Return C_2

Adversary $B^{\textsc{Init},\textsc{New},\textsc{Fn},\textsc{Fin}}$
$A_2^{\textsc{Init},\textsc{New}^*,\textsc{Enc}^*,\textsc{Fin}}$

procedure $\textsc{New}^*$
$v \leftarrow v + 1$; $K_{1,v} \leftarrow\!\!\$\ \mathsf{SE1.KS}$
$\textsc{New}$

procedure $\textsc{Enc}^*(i, N, M, H)$
$C_1 \leftarrow \mathsf{SE1.Enc}(K_{1,i}, N, M, H)$
$x \leftarrow C_1[1..\mathsf{F.il}]$; $P \leftarrow \textsc{Fn}(i, x)$; $Y \leftarrow P \oplus N$
$C_2 \leftarrow Y \| C_1$; Return C_2

Adversary $A_1^{\textsc{Init},\textsc{New},\textsc{Enc},\textsc{Fin}}$
$A_2^{\textsc{Init},\textsc{New}^*,\textsc{Enc}^*,\textsc{Fin}}$

procedure $\textsc{New}^*$
$v \leftarrow v + 1$
$f_v \leftarrow\!\!\$\ \mathrm{FUNC}(\{0,1\}^{\mathsf{F.il}}, \{0,1\}^{\mathsf{F.ol}})$
$\textsc{New}$

procedure $\textsc{Enc}^*(i, N, M, H)$
$C_1 \leftarrow\!\!\$\ \textsc{Enc}(i, N, M, H)$
$x \leftarrow C_1[1..\mathsf{F.il}]$; $P \leftarrow f_i(x)$; $Y \leftarrow P \oplus N$
$C_2 \leftarrow Y \| C_1$; Return C_2

Figure 13: On the left are the games used in proof of Equation (15). $\textsc{Fin}$ is common to all games. On the right are the adversaries for the same proof.

$\mathcal{A}^{\mathrm{ae1}}_{\text{u-y}} \cap \mathcal{A}^{\mathrm{ae1}}_{\mathrm{priv}}$ *and* B *such that*

$$\mathbf{Adv}^{\mathrm{ae2}}_{\mathsf{SE}_{\mathrm{HN1}}}(A_2) \;\le\; \mathbf{Adv}^{\mathrm{ae1}}_{\mathsf{SE1}}(A_1) + \mathbf{Adv}^{\mathrm{prf}}_{\mathsf{F}}(B) + \frac{q_n q_e (q_e - 1)}{2^{\mathsf{F.il}+1}} \;. \tag{15}$$

Adversary A_1 *preserves the resources of* A_2*. Adversary* B *makes* q_n *queries to its* $\textsc{New}$ *oracle and* q_e *queries per user to its* $\textsc{Fn}$ *oracle. Adversary* B *has about the same running time as* A_2*.*

Proof: Adversary A_1 for the authenticity claim is in Figure 12. Adversary A_1's simulation of $\textsc{Enc}$ queries is faithful. We need to check not only Equation (14) but also that A_1 belongs to the claimed class $\mathcal{A}^{\mathrm{auth1}}_{\text{u-y}}$. We claim that when a $\textsc{Vf}$ query of A_2 is winning (accepting and new) in its game, then the corresponding $\textsc{Vf}$ query of A_1 is winning (accepting and new) in its game. This comes down to the following. Fix K_{F} and C_1, let $x = C_1[1..\mathsf{F.il}]$ and let $Y, Y' \in \{0,1\}^{\mathsf{F.ol}}$. Let $N = Y \oplus \mathsf{F.Ev}(K_{\mathsf{F}}, x)$ and $N' = Y' \oplus \mathsf{F.Ev}(K_{\mathsf{F}}, x)$. Then $Y = Y'$ iff $N = N'$. Intuitively, with K_{F}, C_1 fixed, there is a one-to-one correspondence between full ciphertexts $Y \| C_1$ and nonce, core-ciphertext pairs (N, C_1) where $N = Y \oplus \mathsf{F.Ev}(K_{\mathsf{F}}, C_1[1..\mathsf{F.il}])$.

For the proof of privacy, consider the games in Fig. 13. Oracle $\textsc{Dec}$ is dropped, since the privacy adversary makes no queries to it. Game G_0 is the real game. Game G_1 switches from F to random

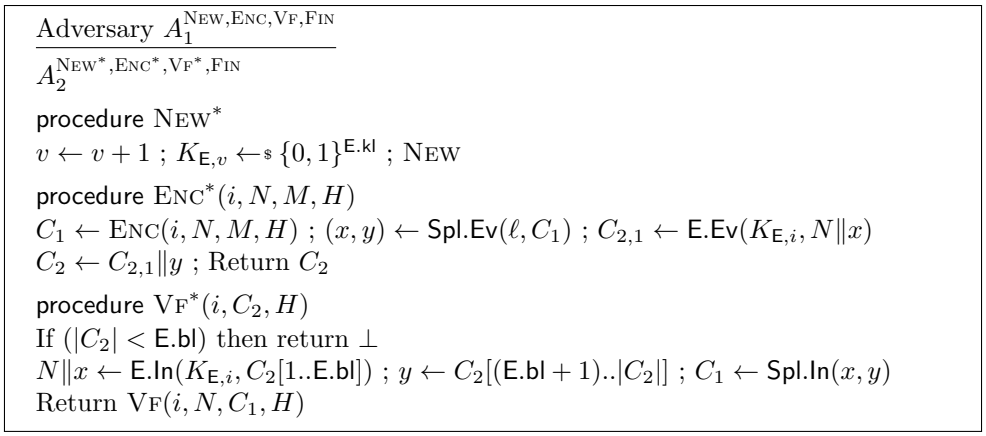

Figure 14: Adversary A_1 used in proving Equations (3) and (8).

functions, which the adversary will not notice due to the assumed PRF security of F. Game G_2 switches to random core ciphertexts, which the adversary will not notice due to the assumed privacy of $\mathsf{SE1}$. Game G_3 switches to random full ciphertexts. Games G_2, G_3 differ only in the boxed code, so that the adversary notices the switch only when two calls to Enc pick the same value of x. This is exactly the probability that bad is set. Proceeding to the details, we have:

$$\begin{aligned}
&\mathbf{Adv}^{\text{ae2}}_{\mathsf{SE}_{\text{HN1}}}(A_2) = \Pr[\text{G}_0(A_2)] - \Pr[\text{G}_3(A_2)] \\
&= (\Pr[\text{G}_0(A_2)] - \Pr[\text{G}_1(A_2)]) \ + \ (\Pr[\text{G}_1(A_2)] - \Pr[\text{G}_2(A_2)]) \ + \ (\Pr[\text{G}_2(A_2)] - \Pr[\text{G}_3(A_2)]) \ .
\end{aligned}$$

Let adversaries A_1 and B be as in Fig. 13. For simplicity we show A_1 as picking f_v at random, but for efficiency (meaning, to keep the running time to the same as that of A_2) this must be implemented via lazy sampling. Then:

$$\begin{aligned}
\Pr[\text{G}_0(A_2)] - \Pr[\text{G}_1(A_2)] &= \mathbf{Adv}^{\text{prf}}_{\mathsf{F}}(B) \ , \\
\Pr[\text{G}_1(A_2)] - \Pr[\text{G}_2(A_2)] &= \mathbf{Adv}^{\text{ae1}}_{\mathsf{SE1}}(A_1) \ , \\
\Pr[\text{G}_2(A_2)] - \Pr[\text{G}_3(A_2)] &\le \Pr[\text{G}_2(A_2) \text{ sets } \mathsf{bad}] \\
&\le \frac{q_n q_e (q_e - 1)}{2^{\mathsf{F.il}+1}} \ .
\end{aligned}$$

The third inequality above used the Fundamental Lemma of Game Playing [17]. Putting the above together yields Equation (15). ▌

D Proof of Theorem 6.2

Proof: Adversary A_1 for the authenticity claim of Equation (3) is in Figure 14.

For the proof of privacy, consider the games in Fig. 15. Oracle Dec is dropped, since the privacy adversary makes no queries to it. Game G_0 is the real game. Game G_1 switches from E to random functions, which the adversary will not notice due to the assumed PRF security of E. Game G_2 switches to random core ciphertexts, which the adversary will not notice due to the assumed privacy of $\mathsf{SE1}$. Game G_2 also has random full ciphertexts due to the uniqueness of nonces. Proceeding to

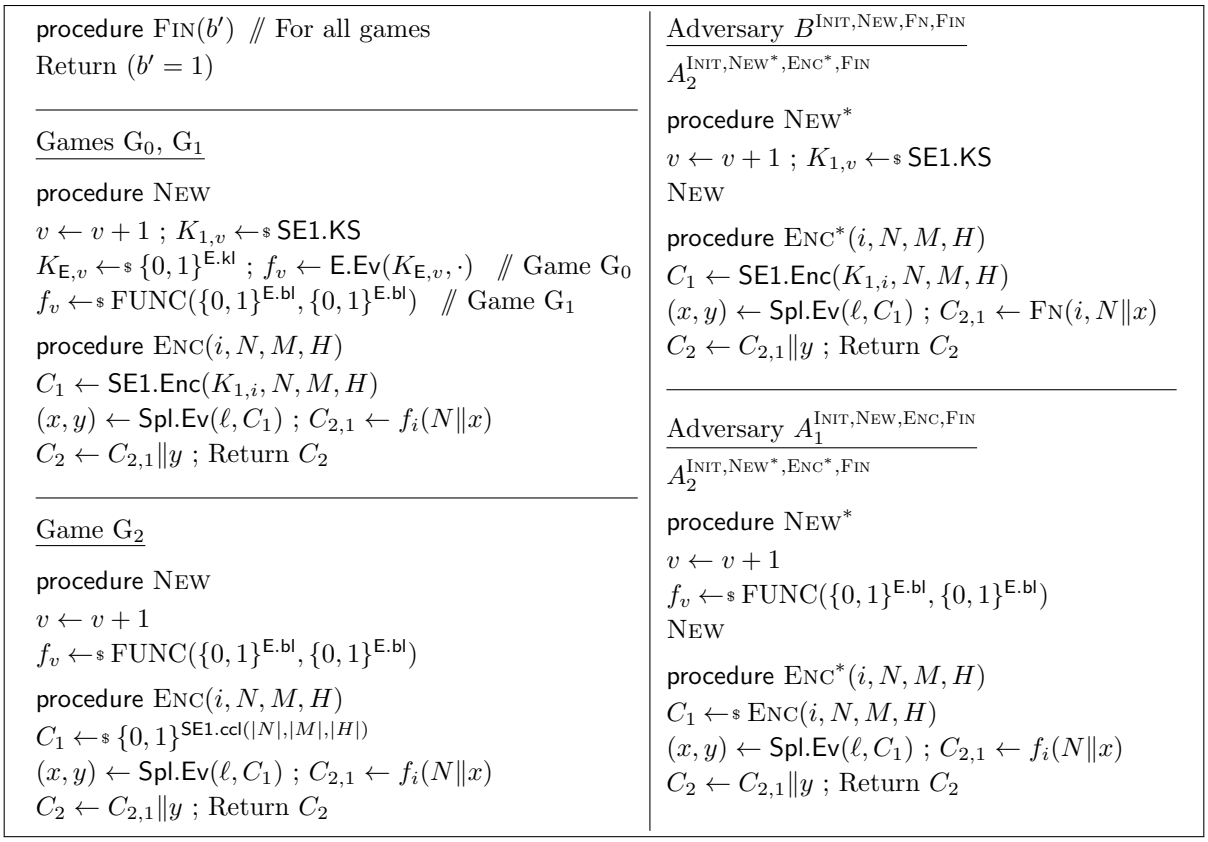

Figure 15: On the left are the games used in proof of Equation (4). G_0, G_1 are also used in the proof of Equation (9). FIN are common to all games. On the right are the adversaries for the same two proofs.

---

the details, we have:

$$\begin{aligned}\mathbf{Adv}^{\mathrm{ae2}}_{\mathsf{SE}_{\mathrm{HN2}}}(A_2) &= \Pr[G_0(A_2)] - \Pr[G_2(A_2)] \\ &= (\Pr[G_0(A_2)] - \Pr[G_1(A_2)]) \ + \ (\Pr[G_1(A_2)] - \Pr[G_2(A_2)]) \ .\end{aligned}$$

Let adversaries A_1 and B be as in Fig. 15. For simplicity we show A_1 as picking f_v at random, but for efficiency (meaning, to keep its running time the same as that of A_2) this must be implemented via lazy sampling. Then:

$$\begin{aligned}\Pr[G_0(A_2)] - \Pr[G_1(A_2)] &= \mathbf{Adv}^{\mathrm{prf}}_{\mathsf{E}}(B) \ , \\ \Pr[G_1(A_2)] - \Pr[G_2(A_2)] &= \mathbf{Adv}^{\mathrm{ae1}}_{\mathsf{SE1}}(A_1) \ .\end{aligned}$$

Putting the above together yields Equation (4). ▌

E Proof of Theorem 6.3

Proof: We assume A_2 does not make trivial queries, meaning it does not make query DEC(i, C_2, H) if it has previously received C_2 in response to an ENC$(i, \cdot, \cdot, H)$ query. Consider the games in Fig. 16.

procedure $\textsc{Fin}(b')$ // For all games
Return $(b' = 1)$

Games G_0, G_1

procedure $\textsc{New}$
$v \leftarrow v + 1$; $K_{1,v} \leftarrow\!\!\$\ \mathsf{SE1.KS}$
$K_{\mathsf{F},v} \leftarrow\!\!\$\ \{0,1\}^{\mathsf{F.kl}}$; $f_v \leftarrow \mathsf{F.Ev}(K_{\mathsf{F},v}, \cdot)$ // Game G_0
$f_v \leftarrow\!\!\$\ \mathrm{FUNC}(\{0,1\}^{\mathsf{F.il}}, \{0,1\}^{\mathsf{F.ol}})$ // Game G_1

procedure $\textsc{Enc}(i, N, M, H)$
$N_1 \leftarrow f_i(N)$; $C_1 \leftarrow \mathsf{SE1.Enc}(K_{1,i}, N_1, M, H)$
Return $N_1 \| C_1$

procedure $\textsc{Dec}(i, C_2, H)$
$N_1 \| C_1 \leftarrow C_2$; $M \leftarrow \mathsf{SE1.Dec}(K_{1,i}, N_1, C_1, H)$
Return M

Game G_2

procedure $\textsc{New}$
$v \leftarrow v + 1$; $K_{1,v} \leftarrow\!\!\$\ \mathsf{SE1.KS}$
$f_v \leftarrow\!\!\$\ \mathrm{FUNC}(\{0,1\}^{\mathsf{F.il}}, \{0,1\}^{\mathsf{F.ol}})$

procedure $\textsc{Enc}(i, N, M, H)$
$N_1 \leftarrow f_i(N)$; $C_1 \leftarrow\!\!\$\ \{0,1\}^{\mathsf{SE1.ccl}(|N_1|,|M|,|H|)}$
Return $N_1 \| C_1$

procedure $\textsc{Dec}(i, C_2, H)$
$M \leftarrow \perp$; Return M

Adversary $B^{\textsc{Init},\textsc{New},\textsc{Fn},\textsc{Fin}}$
$A_2^{\textsc{Init},\textsc{New}^*,\textsc{Enc}^*,\textsc{Dec}^*,\textsc{Fin}}$

procedure $\textsc{New}^*$
$v \leftarrow v + 1$; $K_{1,v} \leftarrow\!\!\$\ \mathsf{SE1.KS}$
$\textsc{New}$

procedure $\textsc{Enc}^*(i, N, M, H)$
$N_1 \leftarrow \textsc{Fn}(i, N)$
$C_1 \leftarrow \mathsf{SE1.Enc}(K_{1,i}, N_1, M, H)$
Return $N_1 \| C_1$

procedure $\textsc{Dec}^*(i, C_2, H)$
$N_1 \| C_1 \leftarrow C_2$
$M \leftarrow \mathsf{SE1.Dec}(K_{1,i}, N_1, C_1, H)$
Return M

Adversary $A_1^{\textsc{Init},\textsc{New},\textsc{Enc},\textsc{Dec},\textsc{Fin}}$
$A_2^{\textsc{Init},\textsc{New}^*,\textsc{Enc}^*,\textsc{Dec}^*,\textsc{Fin}}$

procedure $\textsc{New}^*$
$v \leftarrow v + 1$
$f_v \leftarrow\!\!\$\ \mathrm{FUNC}(\{0,1\}^{\mathsf{F.il}}, \{0,1\}^{\mathsf{F.ol}})$
$\textsc{New}$

procedure $\textsc{Enc}^*(i, N, M, H)$
$N_1 \leftarrow f_i(N)$; $C_1 \leftarrow\!\!\$\ \textsc{Enc}(i, N_1, M, H)$
Return $N_1 \| C_1$

procedure $\textsc{Dec}^*(i, C_2, H)$
$N_1 \| C_1 \leftarrow C_2$; $M \leftarrow \textsc{Dec}^*(i, N_1, C_1, H)$
Return M

Figure 16: On the left are the games used in proof of Equation (5). $\textsc{Fin}$ is common to all games. On the right are the adversaries for the same proof.

Game G_0 is the real game. Game G_1 switches from F to random functions, which the adversary will not notice due to the assumed PRF security of F. Game G_2 switches to random core ciphertexts and $\perp$ replies to $\textsc{Dec}$ queries, which the adversary will not notice due to the assumed AE1$[\mathcal{A}^{\mathrm{ae1}}_{\mathrm{r\text{-}n}}]$-security of $\mathsf{SE1}$. Game G_2 also has random full ciphertexts due to the uniqueness of nonces. Proceeding to the details, we have:

$$\begin{aligned}\mathbf{Adv}^{\mathrm{ae2}}_{\mathsf{SE}_{\mathrm{HN3}}}(A_2) &= \Pr[\mathrm{G}_0(A_2)] - \Pr[\mathrm{G}_2(A_2)] \\ &= (\Pr[\mathrm{G}_0(A_2)] - \Pr[\mathrm{G}_1(A_2)]) \ + \ (\Pr[\mathrm{G}_1(A_2)] - \Pr[\mathrm{G}_2(A_2)]) \ .\end{aligned}$$

Let adversaries A_1 and B be as in Fig. 16. For simplicity we show A_1 as picking f_v at random, but for efficiency (meaning, to keep the running time to the same as that of A_2) this must be implemented via lazy sampling. Adversary A_1 is in the class $\mathcal{A}^{\mathrm{ae1}}_{\mathrm{r\text{-}n}}$ because the nonces it uses in its

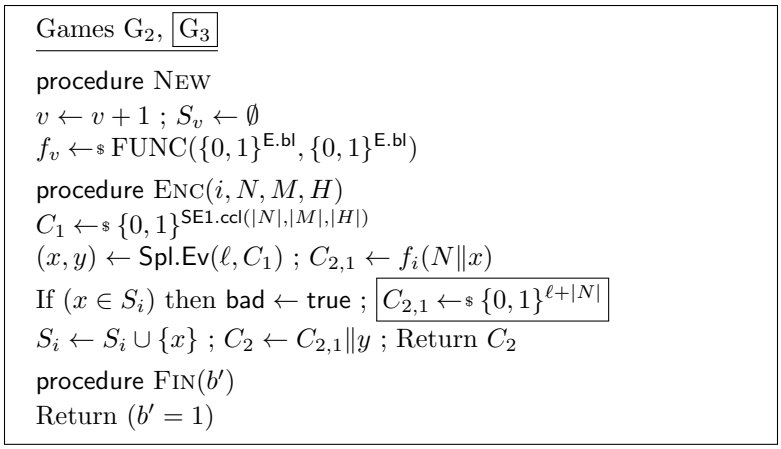

Figure 17: Games G_2, G_3 used in proof of Equation (9).

ENC queries are results of f_i on unique nonces, and are hence random and independent. Then:

$$\Pr[G_0(A_2)] - \Pr[G_1(A_2)] = \mathbf{Adv}^{\mathrm{prf}}_{\mathsf{F}}(B) \ ,$$

$$\Pr[G_1(A_2)] - \Pr[G_2(A_2)] = \mathbf{Adv}^{\mathrm{ae1}}_{\mathsf{SE1}}(A_1) \ .$$

Putting the above together yields Equation (5). ∎

F Proof of Theorem 7.2

Proof: The proof of Theorem 7.2 is very similar to that of Theorem 6.2.

The adversary A_1 used in proving Equation (8) is the same one depicted in Fig. 14. Note that if $A_2 \in \mathcal{A}^{\mathrm{auth2}}_{\text{u-nmh}}$ then $A_1 \in \mathcal{A}^{\mathrm{auth1}}_{\text{u-nmh}}$, meaning that A_1 is in the desired adversary class.

For the proof of privacy, we will make use of games G_0, G_1 from the proof of Theorem 7.2 (Fig. 15), but define the new games G_2, G_3 shown in Fig. 17). As before, G_0 is the real game, while game G_1 switches from E to random functions, which the adversary will not notice due to the assumed PRF security of E. Game G_2 switches to random core ciphertexts, which the adversary will not notice due to the assumed privacy of SE1. Since we can no longer assume that nonces are unique, however, the full ciphertexts may not be random. They will be, however, if the x values do not repeat, allowing us to switch to game G_3 with a loss that is the probability of such a repeat.

Proceeding to the details, assume as usual that A_2 does not make repeat or trivial queries. Then we have

$$\begin{aligned}\mathbf{Adv}^{\mathrm{ae2}}_{\mathsf{SE}_{\mathrm{HN2}}}(A_2) &= \Pr[G_0(A_2)] - \Pr[G_3(A_2)] \\ &= (\Pr[G_0(A_2)] - \Pr[G_1(A_2)]) \ + \ (\Pr[G_1(A_2)] - \Pr[G_2(A_2)]) \ + \ (\Pr[G_2(A_2)] - \Pr[G_3(A_2)]) \ .\end{aligned}$$

To conclude the proof of Equation (9), we have

$$\Pr[G_0(A_2)] - \Pr[G_1(A_2)] = \mathbf{Adv}^{\mathrm{prf}}_{\mathsf{E}}(B) \ ,$$

$$\Pr[G_1(A_2)] - \Pr[G_2(A_2)] = \mathbf{Adv}^{\mathrm{ae1}}_{\mathsf{SE1}}(A_1) \ ,$$

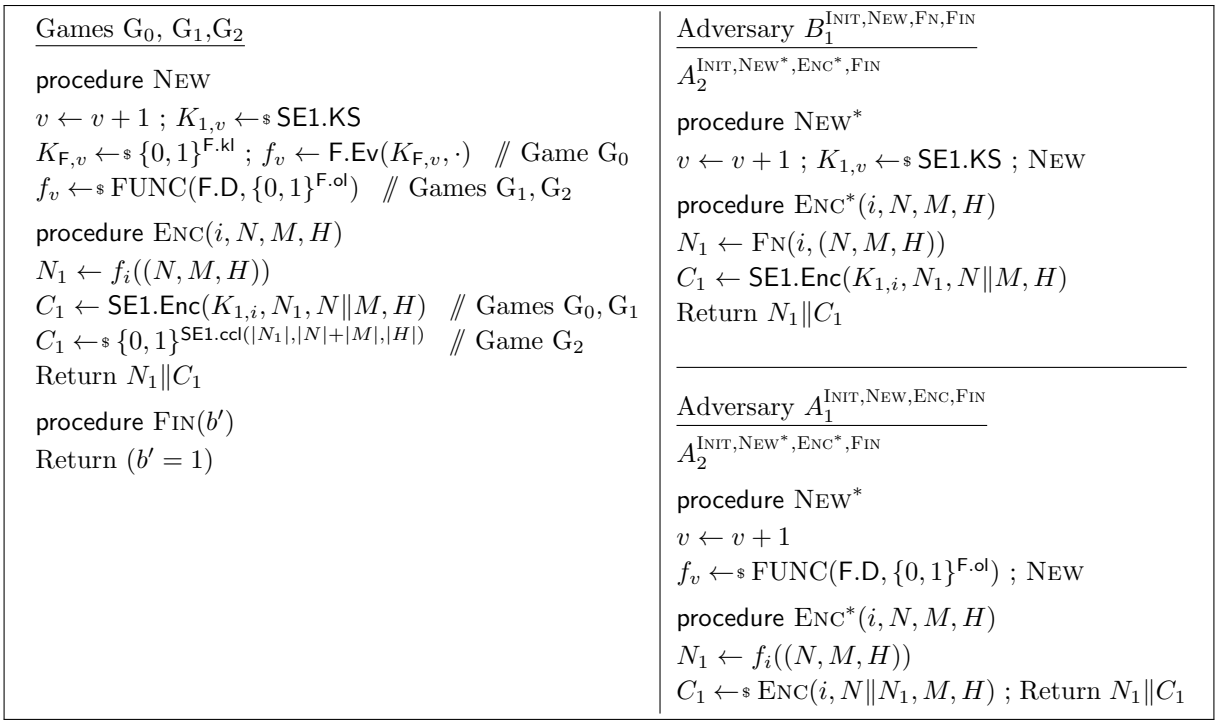

Figure 18: On the left are the games used in proof of Equation (10). On the right are the adversaries for the same proof. Note that $\mathsf{F.D} = \mathsf{SE1.NS} \times \mathsf{SE1.MS} \times \mathsf{SE1.HS}$, as required in th definition of **HN4** in Section 7.

$$\begin{aligned}\Pr[\mathrm{G}_2(A_2)] - \Pr[\mathrm{G}_3(A_2)] &\le \Pr[\mathrm{G}_3(A_2) \text{ sets } \mathsf{bad}] \\ &\le \frac{q_n q_e(q_e-1)}{2^{\ell+1}}\ .\end{aligned} \tag{16}$$

Adversaries B, A_1 for the first two equations above are those depicted in Fig. 15, and now $A_2 \in \mathcal{A}^{\text{ae2}}_{\text{u-nmh}}$ because $A_1 \in \mathcal{A}^{\text{ae1}}_{\text{u-nmh}}$. As before, we assume A_1 implements the f_i via lazy sampling. Games $\mathrm{G}_2, \mathrm{G}_3$ are identical-until-bad, so Equation (16) is by the Fundamental Lemma of Game Playing [17]. ▮

G Proof of Theorem 7.3

Proof: For the proof of privacy, we will make use of the games $\mathrm{G}_0, \mathrm{G}_1, \mathrm{G}_2$ in Fig. 18. Game G_0 is the real game, game G_1 switches to using random functions, which the adversary will not notice due to the assumed PRF security of F, and game G_2 switches to random core ciphertexts. Adversaries B_2, A_1 are also depicted in Fig. 18. Adversary A_2, being a privacy adversary, makes no DEC queries, so we omit giving oracle DEC in the games as well as when it is run by other adversaries. As usual, we assume that A_1 implements f_i using lazy sampling for efficiency. Because we assumed the nonce-message-header triples provided to f_i by A_2 do not repeat, G_2 has random full ciphertexts and $A_1 \in \mathcal{A}^{\text{ae1}}_{\text{r-n}}$. From here, we can derive Equation 10:

$$\mathbf{Adv}^{\text{ae2}}_{\mathsf{SE}_{\text{HN4}}}(A_2) = \Pr[\mathrm{G}_0(A_2)] - \Pr[\mathrm{G}_2(A_2)]$$

Games $\boxed{\mathrm{G}_0}$, G_1

procedure NEW
$v \leftarrow v+1$; $K_{1,v} \leftarrow\!\!\$\ \mathsf{SE1.KS}$
$K_{\mathsf{F},v} \leftarrow\!\!\$\ \{0,1\}^{\mathsf{F.kl}}$; $f_v \leftarrow \mathsf{F.Ev}(K_{\mathsf{F},v}, \cdot)$ // Game G_0
$f_v \leftarrow\!\!\$\ \mathrm{FUNC}(\{0,1\}^{\mathsf{F.il}}, \{0,1\}^{\mathsf{F.ol}})$ // Game G_1

procedure ENC(i, N, M, H)
$N_1 \leftarrow f_i((N, M, H))$
$C_1 \leftarrow \mathsf{SE1.Enc}(K_{1,i}, N_1, N\|M, H)$
Return $N_1\|C_1$

procedure VF(i, C_2, H)
$N_1\|C_1 \leftarrow C_2$; $X \leftarrow \mathsf{SE1.Dec}(K_{1,i}, N_1, C_1, H)$
If $(X = \perp)$ then return false
$N\|M \leftarrow X$; $T \leftarrow f_i((N, M, H))$
If $(T = N_1)$ then win $\leftarrow$ true
Return $(T = N_1)$

procedure FIN
Return win

Adversary $B_2^{\text{INIT,NEW,FN,FIN}}$

$A_2^{\text{INIT,NEW}^*,\text{ENC}^*,\text{VF}^*,\text{FIN}^*}$

procedure NEW*
$v \leftarrow v+1$; $K_{1,v} \leftarrow\!\!\$\ \mathsf{SE1.KS}$; NEW

procedure ENC$^*(i, N, M, H)$
$N_1 \leftarrow \mathrm{FN}(i, (N, M, H))$
$C_1 \leftarrow \mathsf{SE1.Enc}(K_{1,i}, N_1, N\|M, H)$
Return $N_1\|C_1$

procedure VF$^*(i, C_2, H)$
$N_1\|C_1 \leftarrow C_2$
$X \leftarrow \mathsf{SE1.Dec}(K_{1,i}, N_1, C_1, H)$
If $(X = \perp)$ then return false
$N\|M \leftarrow X$; $T \leftarrow \mathrm{FN}(i, (N, M, H))$
If $(T = N_1)$ then win $\leftarrow$ true
Return $(T = N_1)$

procedure FIN*
If win = true then $b' \leftarrow 1$ else $b' \leftarrow 0$
FIN(b')

Figure 19: On the left are the games used in proof of Equation (11). On the right is the adversary for the same proof.

$$= (\Pr[\mathrm{G}_0(A_2)] - \Pr[\mathrm{G}_1(A_2)]) \ + \ (\Pr[\mathrm{G}_1(A_2)] - \Pr[\mathrm{G}_2(A_2)])$$
$$= \mathbf{Adv}^{\mathrm{prf}}_{\mathsf{F}}(B) \ + \ \mathbf{Adv}^{\mathrm{ae1}}_{\mathsf{SE}_{\mathsf{SE1}}}(A_1) \ .$$

Now we proceed to the authenticity proof. As before, we assume that A_2 does not make repeat or trivial queries. Games $\mathrm{G}_0, \mathrm{G}_1$ and adversary B_2 are depicted in Fig. 19. As before, the difference is that G_1 switches the f_v functions to random. We have

$$\begin{aligned}\mathbf{Adv}^{\mathrm{auth2}}_{\mathsf{SE}_{\mathsf{SE}_{\mathrm{HN4}}}}(A_2) &= \Pr[\mathrm{G}_0(A_2)] \\ &= \Pr[\mathrm{G}_1(A_2)] + (\Pr[\mathrm{G}_0(A_2)] - \Pr[\mathrm{G}_1(A_2)]) \ .\end{aligned}$$

To complete the proof, we claim that

$$\begin{aligned}\Pr[\mathrm{G}_0(A_2)] - \Pr[\mathrm{G}_1(A_2)] &\le \mathbf{Adv}^{\mathrm{prf}}_{\mathsf{F}}(B_2) \\ \Pr[\mathrm{G}_1(A_2)] &\le \frac{q_n q_v}{2^{\mathsf{SE1.nl}}} \ . \qquad (17)\end{aligned}$$

Equation (17) is due to the assumed tidiness of SE1, as follows. Suppose $\perp \neq X = N\|M$. Tidiness plus the assumption that A_2 makes no trivial queries say that (i, N, M, H) was not a prior query to ENC, which means that $T = N_1$ with probability at most $2^{-\mathsf{SE1.nl}}$. ∎

Adversary $B_1^{\text{Init},\text{New},\text{Fn},\text{Fin}}$	Adversary $B_2^{\text{Init},\text{New},\text{Fn},\text{Fn}^{-1},\text{Fin}}$
$A^{\text{Init},\text{New},\text{Enc}^*,\text{Fin}}$	Init $A_2^{\text{New},\text{Enc}^*,\text{Vf}^*,\text{Fin}^*}$
procedure $\text{Enc}^*(i, N, M, H)$ $C_2 \leftarrow \text{Fn}(i, H, 0^{\ell_z} \| N \| M)$ Return C_2	procedure $\text{Enc}^*(i, N, M, H)$ $C_2 \leftarrow \text{Fn}(i, H, 0^{\ell_z} \| N \| M)$; Return C_2
	procedure $\text{Vf}^*(i, C_2, H)$ $X \leftarrow \text{Fn}^{-1}(i, H, C_2)$ If $(X[1..\ell_z] \neq 0^{\ell_z})$ then return false Else win $\leftarrow$ true ; Return true
	procedure Fin^* If (win = true) then $b' \leftarrow 1$ else $b' \leftarrow 0$ $\text{Fin}(b')$

Figure 20: Adversaries used in the proof of Theorem 7.4.

H Proof of Theorem 7.4

Proof: Adversary B_1 referred to in Equation (12) is in Fig. 20. Init, Fin and New are all unchanged, and Enc is simulated as shown. Since A is a privacy adversary, we do not need to simulate a decryption oracle.

Adversary B_2 referred to in Equation (13) is also presented in Fig. 20. As before, we assume A_2 neither makes repeat encryption or verification queries, nor makes trivial verification queries, meaning it does not make query $\text{Vf}(i, C_2, H)$ if it has previously received C_2 in response to an $\text{Enc}(i, \cdot, \cdot, H)$ query and also $|C_2| \geq \ell + \ell_z$ in any $\text{Vf}(i, C_2, H)$ query.

Let b be the challenge bit of game $\mathbf{G}^{\text{prp-cca}}_{\mathsf{TE}}$ and let b' be the bit that B_2 queries to $\mathbf{G}^{\text{prp-cca}}_{\mathsf{TE}}$.Fin. Then,

$$\mathbf{Adv}^{\text{prp-cca}}_{\mathsf{TE}}(B_2) = \Pr[\, b' = 1 \,|\, b = 1 \,] - \Pr[\, b' = 1 \,|\, b = 0 \,] \,.$$

To complete the proof, we claim that

$$\Pr[\, b' = 1 \,|\, b = 1 \,] \geq \mathbf{Adv}^{\text{auth2}}_{\mathsf{SE}_{\text{HN5}}}(A_2) \tag{18}$$

$$\Pr[\, b' = 1 \,|\, b = 0 \,] \leq \frac{2 q_n q_v}{2^{\ell_z}} \,. \tag{19}$$

Note that $b' = 1$ if and only if some query of A_2 to Vf^* returns true. If $b = 1$ then this happens if A_2 wins $\mathbf{G}^{\text{auth2}}_{\mathsf{SE}_{\text{HN5}}}$, justifying Equation (18). Now suppose $b = 0$. Consider a particular user i and the j-th Vf query to that user. Let C_2 be the ciphertext in that query and assume s queries to Enc have been made to user i prior to this Vf query. Then the probability that this Vf query sets win to true is at most

$$\begin{aligned} \frac{2^{|C_2|-\ell_z} - s}{2^{|C_2|} - (s + j - 1)} &\leq \frac{2^{|C_2|-\ell_z}}{2^{|C_2|} - (q_e + q_v - 1)} \\ &= \frac{1}{2^{\ell_z}} \cdot \frac{1}{1 - (q_e + q_v - 1) \cdot 2^{-|C_2|}} \,. \end{aligned}$$

Games G_0, $\boxed{\mathrm{G}_1}$, G_2, G_3

procedure NEW
$K \leftarrow\!\!\$\ \{0,1\}^{\mathsf{E.bl}}$; $\pi \leftarrow \mathsf{E.Ev}(K,\cdot)$ // Games $\mathrm{G}_0, \mathrm{G}_1, \mathrm{G}_2$
$\pi \leftarrow\!\!\$\ \mathrm{FUNC}(\{0,1\}^{\mathsf{E.bl}})$ // Game G_3
$L \leftarrow \pi(0^{\mathsf{E.bl}})$

procedure ENC$(1, N, M, H)$
$i \leftarrow i+1$; $N_i \leftarrow N$; $M_i \leftarrow M$; $H_i \leftarrow H$; $P_i \leftarrow \mathrm{Pad}^{\pi}_{2,\ell+|M|}(N)$; $M^* \leftarrow 0^{\ell}\|M$; $C^*_{1,i} \leftarrow M \oplus P_i$
$h_i \leftarrow \mathsf{H.Ev}(L, (C^*_{1,i}, H_i))$; $p_i \leftarrow \pi(N_i\|\langle 1\rangle_{\mathsf{E.bl}-\ell})$; $\mathcal{P} \leftarrow \mathcal{P} \cup \{p_i\}$; $\mathcal{N} \leftarrow \mathcal{N} \cup \{N_i\|\langle 1\rangle_{\mathsf{E.bl}-\ell}\}$
$\tau_i \leftarrow h_i \oplus p_i$; Return $\tau_i\|C^*_{1,i}$

procedure VF$(1, C_2, H)$ // Games G_0, $\boxed{\mathrm{G}_1}$
$\tau\|C^*_1 \leftarrow C_2$; $h \leftarrow \mathsf{H.Ev}(L, (C^*_1, H))$; $p \leftarrow \tau \oplus h$
If $(p \in \mathcal{P})$ then $\mathsf{bad} \leftarrow \mathsf{true}$; $\boxed{p \leftarrow\!\!\$\ \{0,1\}^{\mathsf{E.bl}} \setminus \mathcal{P}}$
$y \leftarrow \pi^{-1}(p)$; $N\|w \leftarrow y$; $P \leftarrow \mathrm{Pad}^{\pi}_{2,|C^*_1|}(N)$; $M^* \leftarrow C^*_1 \oplus P$; $x\|M \leftarrow M^*$
$\mathsf{win} \leftarrow (x = 0^{\ell})$ and $(w = \langle 1\rangle_{\mathsf{E.bl}-\ell})$; Return false

procedure VF$(1, C_2, H)$ // Games $\mathrm{G}_2, \mathrm{G}_3$
$\tau\|C^*_1 \leftarrow C_2$; $y \leftarrow\!\!\$\ \{0,1\}^{\mathsf{E.bl}} \setminus \mathcal{N}$; $N\|w \leftarrow y$
$P \leftarrow \mathrm{Pad}^{\pi}_{2,|C^*_1|}(N)$; $M^* \leftarrow C^*_1 \oplus P$; $x\|M \leftarrow M^*$
$\mathsf{win} \leftarrow (x = 0^{\ell})$ and $(w = \langle 1\rangle_{\mathsf{E.bl}-\ell})$; Return false

procedure FIN
Return win

Figure 21: First set of games used in proof of Theorem 8.2. Next to procedure names, we indicate the games to which they belong. Unannotated procedures belong to all games in the Figure.

But $|C_2| \geq \ell + \ell_z$ for any ciphertext C_2 in a VF query, and we assumed $q_e + q_v \leq 2^{\ell+\ell_z-1}$, so, across all queries, the probability that win is set to true is at most

$$\frac{q_n q_v}{2^{\ell_z}} \cdot \frac{1}{1-(q_e+q_v-1)\cdot 2^{-(\ell+\ell_z)}} \leq \frac{q_n q_v}{2^{\ell_z}} \cdot \frac{1}{1-2^{(\ell+\ell_z-1)}\cdot 2^{-(\ell+\ell_z)}}$$
$$= \frac{q_n q_v}{2^{\ell_z}} \cdot \frac{1}{1-2^{-1}} ,$$

which yields Equation (19). ∎

I Proof of Theorem 8.2

Proof: Consider the games of Figure 21. We claim that:

$$\mathbf{Adv}^{\mathrm{auth2}}_{\mathsf{SE2}}(A_2) = \Pr[\mathrm{G}_0(A_2)] \tag{20}$$
$$= \Pr[\mathrm{G}_1(A_2)] + (\Pr[\mathrm{G}_0(A_2)] - \Pr[\mathrm{G}_1(A_2)])$$
$$\leq \Pr[\mathrm{G}_1(A_2)] + \Pr[\mathrm{G}_1(A_2) \text{ sets } \mathsf{bad}] . \tag{21}$$

Let us now explain games $\mathrm{G}_0, \mathrm{G}_1$ and justify the above. Adversary A_2, by assumption, makes a single query to its NEW oracle, initializing the single user under consideration. Our games pick,

for this user, a key K for E, and let L be the corresponding key for H. The adversary then makes q_e queries to ENC. Since all are directed at user 1, we hardwire 1 as the first input to the oracle, and can think of the adversary queries as triples $(N_1, M_1, H_1), , \ldots, (N_{q_e}, M_{q_e}, H_{q_e})$. The games compute replies correctly according to the encryption algorithm of the scheme. Its ENC queries completed, the adversary makes its single DEC query, which we view as a pair (C_2, H), hardwiring the user number 1 in the oracle. What is returned to the adversary as response does not matter, since the only further action of the adversary is its mandated call to FIN(), and accordingly all our games return false in reply to the DEC query. But internally the games set the win flag, and its value is what FIN() returns as the game output. We assume the adversary's DEC query is non-trivial, meaning $(\tau \| C_1^*, H) \notin \{\, (\tau_i \| C_{1,i}^*, H_i) \;:\; 1 \le i \le q_e \,\}$. Game G_0 excludes the boxed code, and thus sets win correctly, justifying Equation (20). We will get to the meaning of the boxed code later; for now what matters is that, games $\mathrm{G}_0, \mathrm{G}_1$ being identical-until-bad, the Fundamental Lemma of Game Playing [17] justifies Equation (21). This leaves us with two tasks: (1) to bound $\Pr[\mathrm{G}_1(A_2)]$ and (2) to bound $\Pr[\mathrm{G}_1(A_2)$ sets $\mathsf{bad}]$.

We start with (1). Game G_2 changes only procedure VF, which, rather than setting $y \leftarrow \pi^{-1}(p)$, picks y at random from $\{0,1\}^{\mathsf{E.bl}} \setminus \mathcal{N}$. We claim this does not change the probability of winning, meaning

$$\Pr[\mathrm{G}_1(A_2)] = \Pr[\mathrm{G}_2(A_2)] \;. \tag{22}$$

The justification of Equation (22) is that in game G_1, the point p is chosen uniformly at random from $\{0,1\}^{\mathsf{E.bl}} \setminus S$, and π is a permutation, so $y \leftarrow \pi^{-1}(p)$ is distributed uniformly at random in $\{0,1\}^{\mathsf{E.bl}} \setminus \mathcal{N}$. Note that this claim does not rely on any security property of, or security assumption about, the blockcipher E, but only on the fact that $\pi = \mathsf{E.Ev}(K, \cdot)$ is a permutation, which can be regarded as fixed in this argument.

Game G_3 switches π from $\mathsf{E.Ev}(K, \cdot)$ to a random function, the change being in procedure NEW alone, and we have

$$\Pr[\mathrm{G}_2(A_2)] = \Pr[\mathrm{G}_3(A_2)] + (\Pr[\mathrm{G}_2(A_2)] - \Pr[\mathrm{G}_3(A_2)]) \;.$$

It is now easy to build a prf-adversary B_0 such that

$$\Pr[\mathrm{G}_2(A_2)] - \Pr[\mathrm{G}_3(A_2)] \le \mathbf{Adv}^{\mathrm{prf}}_{\mathsf{E}}(B_0) \;.$$

The design of B_0 is standard and we omit the details, but we note that the elimination of the computation of π^{-1} was important to be able to rely only on prf security of E, rather than needing to make the stronger assumption that E is prp-cca (also called strong prp) secure.

We are now in a position to exploit the 0^ℓ redundancy that our scheme adds to the message. We claim that

$$\Pr[\mathrm{G}_3] \le \frac{1}{2^\ell} \;. \tag{23}$$

To justify Equation (23), we first claim that if game G_3 returns true then $N \notin \{N_1, \ldots, N_{q_e}\}$. If so (we will justify this claim in a bit), π is being invoked on new points (ones to which it has not been already applied in ENC queries) in the computation $P \leftarrow \mathrm{Pad}^{\pi}_{2,|C_1^*|}(N)$, yielding Equation (23). Returning to the claim, assume game G_3 returns true. Then it must be that $w = \langle 1 \rangle_{\mathsf{E.bl}-\ell}$. Assume towards a contradiction that $N = N_i$ for some i. Then $y = N \| w = N_i \| \langle 1 \rangle_{\mathsf{E.bl}-\ell}$, putting y in $\mathcal{N}$, but y was drawn from outside $\mathcal{N}$, which is the desired contradiction establishing the claim.

Games G_4, G_5, G_6

procedure NEW
$K \leftarrow\!\!\$\ \{0,1\}^{\mathsf{E.bl}}$; $\pi \leftarrow \mathsf{E.Ev}(K,\cdot)$ // Games $\mathrm{G}_4, \mathrm{G}_5$
$\pi \leftarrow\!\!\$\ \mathrm{FUNC}(\{0,1\}^{\mathsf{E.bl}}, \{0,1\}^{\mathsf{E.bl}})$ // Game G_6
$L \leftarrow \pi(0^{\mathsf{E.bl}})$

procedure ENC$(1, N, M, H)$
$i \leftarrow i+1$; $N_i \leftarrow N$; $M_i \leftarrow M$; $H_i \leftarrow H$; $P_i \leftarrow \mathrm{Pad}^{\pi}_{2,\ell+|M|}(N)$; $M^* \leftarrow 0^{\ell} \| M$; $C^*_{1,i} \leftarrow M \oplus P_i$
$h_i \leftarrow \mathsf{H.Ev}(L, (C^*_{1,i}, H_i))$; $p_i \leftarrow \pi(N_i \| \langle 1 \rangle_{\mathsf{E.bl}-\ell})$; $\mathcal{P} \leftarrow \mathcal{P} \cup \{p_i\}$; $\tau_i \leftarrow h_i \oplus p_i$; Return $\tau_i \| C^*_{1,i}$

procedure VF$(1, C_2, H)$
$\tau \| C^*_1 \leftarrow C_2$; $h \leftarrow \mathsf{H.Ev}(L, (C^*_1, H))$; $p \leftarrow \tau \oplus h$; Return false

procedure FIN // Game G_4
Return $(p \in \mathcal{P})$

procedure FIN // Games $\mathrm{G}_5, \mathrm{G}_6$
Return $(\exists\, i : (\,(h \oplus h_i = \tau \oplus \tau_i) \text{ and } (C^*_1, H) \neq (C^*_{1,i}, H_i)\,))$

Game G_7

procedure NEW
$L \leftarrow\!\!\$\ \{0,1\}^{\mathsf{E.bl}}$

procedure ENC$(1, N, M, H)$
$i \leftarrow i+1$; $H_i \leftarrow H$; $C^*_{1,i} \leftarrow\!\!\$\ \{0,1\}^{\ell+|M|}$; $\tau_i \leftarrow\!\!\$\ \{0,1\}^{\mathsf{E.bl}}$
Return $\tau_i \| C^*_{1,i}$

procedure VF$(1, C_2, H)$
$\tau \| C^*_1 \leftarrow C_2$; $h \leftarrow \mathsf{H.Ev}(L, (C^*_1, H))$; Return false

procedure FIN
For $j = 1, \ldots, i$ do $h_i \leftarrow \mathsf{H.Ev}(L, (C^*_{1,i}, H_i))$
Return $(\exists\, i : (\,(h \oplus h_i = \tau \oplus \tau_i) \text{ and } (C^*_1, H) \neq (C^*_{1,i}, H_i)\,))$

Adversary $C^{\mathrm{INIT},\mathrm{FIN}}$

INIT
$A_2^{\mathrm{NEW}^*, \mathrm{ENC}^*, \mathrm{VF}^*, \mathrm{FIN}^*}$

procedure NEW*
Return

procedure ENC$^*(1, N, M, H)$
$i \leftarrow i+1$; $H_i \leftarrow H$
$C^*_{1,i} \leftarrow\!\!\$\ \{0,1\}^{\ell+|M|}$; $\tau_i \leftarrow\!\!\$\ \{0,1\}^{\mathsf{E.bl}}$
Return $\tau_i \| C^*_{1,i}$

procedure VF$^*(1, C_2, H)$
$\tau \| C^*_1 \leftarrow C_2$; Return false

procedure FIN*
$j \leftarrow\!\!\$\ \{1, \ldots, q_e\}$
FIN$(\,(C^*_1, H), (C^*_{1,j}, H_j)\,)$

Figure 22: On the top are further games used in the proof of Theorem 8.2. Lines in code, or procedure names, may be annotated with the names of games which include them, procedures whose names are unannotated belonging to all games. On the bottom left is a final game and on the bottom right is the axu-adversary.

Putting the above together, we have now shown that

$$\Pr[\mathrm{G}_1(A_2)] \leq \mathbf{Adv}^{\mathrm{prf}}_{\mathsf{E}}(B_0) + \frac{1}{2^{\ell}} \,. \tag{24}$$

Next we give adversaries B_1, C such that

$$\Pr[\mathrm{G}_1(A_2) \text{ sets } \mathsf{bad}] \leq \mathbf{Adv}^{\mathrm{prf}}_{\mathsf{E}}(B_1) + q_e \cdot \mathbf{Adv}^{\mathrm{axu}}_{\mathsf{H}}(C) \,. \tag{25}$$

For this, consider the games of Figure 22. We claim

$$\Pr[\mathrm{G}_1(A_2) \text{ sets } \mathsf{bad}] = \Pr[\mathrm{G}_4(A_2)] \tag{26}$$

$$= \Pr[\mathrm{G}_5(A_2)] \; . \tag{27}$$

Game G_4 results from moving the condition setting bad in G_3 to $\textsc{Fin}()$ and dropping unused code, justifying Equation (26). To justify Equation (27), we show that if $p \notin \mathcal{P}$ then there exists i such that $(C_1^*, H) \neq (C_{1,i}^*, H_i)$ but $h \oplus \tau = h_i \oplus \tau_i$, meaning there is a (non-trivial) xor computed for $\mathsf{H.Ev}(L, \cdot)$. That $p \in \mathcal{P}$ means there is some i such that $p = p_i$. (This i need not be unique.) So $h \oplus \tau = h_i \oplus \tau_i$. Now assume towards a contradiction that $(C_1^*, H) = (C_{1,i}^*, H_i)$. Since $h = \mathsf{H.Ev}(L, (C_1^*, H))$ and $h_i = \mathsf{H.Ev}(L, (C_{1,i}^*, H_i))$, we get $h = h_i$. But we already had $h \oplus \tau = h_i \oplus \tau_i$, so we have $\tau = \tau_i$. This means $(\tau \| C_1^*, H) = (\tau_i \| C_{1,i}^*, H_i)$, which contradicts the assumption that the $\textsc{Dec}$ query of the adversary is non-trivial. This concludes the justification of Equation (27).

Game G_6 switches π from $\mathsf{E.Ev}(K, \cdot)$ to a random function, the change being only in $\textsc{New}$, and we have

$$\Pr[\mathrm{G}_5(A_2)] = \Pr[\mathrm{G}_6(A_2)] + (\Pr[\mathrm{G}_5(A_2)] - \Pr[\mathrm{G}_6(A_2)]) \; .$$

Now we can design adversary B_1 such that

$$\Pr[\mathrm{G}_5(A_2)] - \Pr[\mathrm{G}_6(A_2)] \leq \mathbf{Adv}^{\mathrm{prf}}_{\mathsf{E}}(B_1) \; . \tag{28}$$

The design of B_1 is standard and omitted. With π a random function in G_6, the hash key L, and the ciphertexts returned in G_6 in response to $\textsc{Enc}$ queries, are random, so game G_7 directly picks them that way. This allows it to delay computing the hashes to $\textsc{Fin}()$. We have

$$\Pr[\mathrm{G}_6(A_2)] = \Pr[\mathrm{G}_7(A_2)] \; .$$

The bottom right of Figure 22 shows our axu-adversary C. It runs A_2, responding to $\textsc{Enc}$ queries with random strings, as per game G_7. It returns, as its two messages, the hash-input for the $\textsc{Vf}$ query, and a random one of the q_e hash-inputs for the $\textsc{Enc}$ queries. We have

$$\mathbf{Adv}^{\mathrm{axu}}_{\mathsf{H}}(C) \geq \frac{1}{q_e} \cdot \Pr[\mathrm{G}_7(A_2)] \; . \tag{29}$$

Putting the above together we have Equation (25).

At this point we have shown

$$\mathbf{Adv}^{\mathrm{auth2}}_{\mathsf{SE2}}(A_2) \leq \mathbf{Adv}^{\mathrm{prf}}_{\mathsf{E}}(B_0) + \mathbf{Adv}^{\mathrm{prf}}_{\mathsf{E}}(B_1) + q_e \cdot \mathbf{Adv}^{\mathrm{axu}}_{\mathsf{H}}(C) + \frac{1}{2^\ell} \; . \tag{30}$$

We merge B_0, B_1 into a single adversary B as follows. Let B pick $c \leftarrow\!\!\$\, \{0,1\}$ and run B_c. Then

$$\mathbf{Adv}^{\mathrm{prf}}_{\mathsf{E}}(B) = \frac{1}{2} \cdot \mathbf{Adv}^{\mathrm{prf}}_{\mathsf{E}}(B_0) + \frac{1}{2} \cdot \mathbf{Adv}^{\mathrm{prf}}_{\mathsf{E}}(B_1) \; . \tag{31}$$

Putting together Equations (30) and (31) concludes the proof. ∎