Reconsidering Generic Composition

Chanathip Namprempre[1] and Phillip Rogaway[2] and Thomas Shrimpton[3]

[1] Dept. of Electrical and Computer Engineering, Thammasat University, Thailand
[2] Dept. of Computer Science, University of California, Davis, USA
[3] Dept. of Computer Science, Portland State University, Portland, USA

March 19, 2014

Abstract. In the context of authenticated encryption (AE), *generic composition* has referred to the construction of an AE scheme by gluing together a conventional (privacy-only) encryption scheme and a MAC. Since the work of Bellare and Namprempre (2000) and then Krawczyk (2001), the conventional wisdom has become that there are three forms of generic composition, with Encrypt-then-MAC the only one that generically works. However, many caveats to this understanding have surfaced over the years. Here we explore this issue further, showing how this understanding oversimplifies the situation because it ignores the results' sensitivity to definitional choices. When encryption is formalized differently, making it either IV-based or nonce-based, rather than probabilistic, and when the AE goal is likewise changed to take in a nonce, qualitatively different results emerge. We explore these alternatives versions of the generic-composition story. We also evidence the overreaching understanding of prior generic-composition results by pointing out that the Encrypt-then-MAC mechanism of ISO 19772 is completely wrong.

Keywords: authenticated encryption, generic composition, IV-based encryption, nonce-based encryption.

Table of Contents

1 Introduction

Specificity of GC results. We revisit the problem of creating an authenticated encryption (AE) scheme by generic composition (GC). This well-known problem was first articulated and studied in a paper by Bellare and Namprempre [3, 4] (henceforth BN). A review of discourse surrounding BN makes clear that, to its readers, the paper's message was that

1. there are **three ways** to glue together a (privacy-only) encryption scheme and a MAC, well summarized by the names Encrypt-and-MAC, Encrypt-then-MAC, and MAC-then-Encrypt;
2. but of these three ways, only **Encrypt-then-MAC** works well: it alone will always be secure when the underlying primitives are sound.

While BN does of course contain such results, we claim that the understanding articulated above is nonetheless off-base, for it makes no reference to the *type* of schemes from which one starts, nor the *type* of scheme one aims to build. The omission is untenable because GC results turn out to depend crucially on these choices—and multiple alternatives are as reasonable as those selected by BN.

Types of encryption schemes. What are these definitional choices allegedly so important for GC? See Figure 1. To begin, in schemes for **probabilistic encryption** (pE), the encryption algorithm is provided a key and plaintext, and, by a process that employs internal coins, it generates a ciphertext [2, 13]. The plaintext must be recoverable from (just) the ciphertext and key. Syntactically, a **probabilistic authenticated-encryption** (pAE) scheme is the same as a pE scheme. But a pAE scheme should also protect the receiver against *forged* ciphertexts [3–5].

BN focuses on turning a pE scheme and a MAC into a pAE scheme. But conventional, standardized encryption schemes—modes like CBC or CTR [9, 12]—are not really pE schemes, for in lieu of internally generated random coins they use an externally provided IV (initialization vector). Let us call such schemes **IV-based encryption** (ivE). When security is proven for such schemes [1, 2] the IV is selected uniformly at random, and then, for definitional purposes, prepended to the ciphertext. But the standards do not insist that the IV be uniform, nor do they consider it to be part of the ciphertext [9, 12, 14]. In practice, IVs are frequently non-random or communicated out-of-band. In effect, theorists have considered the pE scheme canonically induced by an ivE scheme—but the two objects are not the same thing.

A scheme for **nonce-based encryption** (nE) is syntactically similar to an ivE scheme. Again there is an externally provided value, like the IV, but now referred to as a *nonce* ("number used once"). Security for nE is expected to hold as long as the nonce is not repeated [20]. One expects ease-of-correct-use advantages over ivE, insofar as it should be easier for a user to successfully provide a non-repeating value than a random IV. Standard ivE schemes that are secure when the IV is random (eg, CBC or CTR) are not secure in the nE sense: they are easily attacked if the IV is merely a nonce.

Finally, a scheme for **nonce-based authenticated-encryption** (nAE) is like an nE scheme but the decrypting party should reject illegitimate ciphertexts. Standardized AE methods—modes like CCM, GCM, and OCB [10, 11, 15]—are secure as nAE schemes. Following standard practice, nAE schemes are further assumed to include *associated data* (AD). This string, provided to the encryption and decryption algorithms, is authenticated but not encrypted [19]. For practical utility of AE, the AD turns out to be crucial.

type	$\mathcal{E}$ takes	$\mathcal{D}$ takes	**summary of basic security requirement**
pE	K, M	K, C	privacy: ind = (ciphertexts $\approx \mathcal{E}_K$(rand-bits)) – $\mathcal{E}_K$ flips the needed coins
pAE	K, M	K, C	privacy + auth: ind, plus adv can't forge ciphertexts – $\mathcal{E}_K$ flips the needed coins
ivE	K, IV, M	K, IV, C	privacy: $(IV_i \parallel C_i) \approx$ rand-bits – the environment selects a random IV
nE	K, N, M	K, N, C	privacy: ind\$ = (ciphertexts $\approx$ rand-bits) – adv provides non-repeating N
nAE	K, N, A, M	K, N, A, C	privacy + auth: ind\$ + adv can't forge ciphertexts – adv selects non-repeating N

Fig. 1. Types of AE schemes. The first column gives the name we will use for this type of symmetric encryption scheme. The second and third columns specify the inputs to encryption $\mathcal{E}$ and decryption $\mathcal{D}$: the key K, plaintext M, ciphertext C, initialization vector IV, nonce N, and associated data A. The final column gives a brief description of the main security definition we will use.

Contributions. This paper explores how GC results turn on the basic definitional distinctions named above. Consider the GC scheme of ISO 19772 [15]. The scheme is in the Encrypt-then-MAC tradition, and the standard appeals to BN to support this choice [15, p. 15]. Yet the ISO scheme is wrong. (It is currently being revised in response to our critique [17].) The root problem, we maintain, is that the standard attends to none of the distinctions just described. To apply BN's Encrypt-then-MAC result to a scheme like CBC one would need to select its IV uniformly at random, prepend this to the ciphertext, then take the MAC over this string. But the ISO standard does none of this; the IV is not required to be random, and the scope of the MAC doesn't include it. This makes the scheme trivial to break. A discussion of the ISO scheme appears in Section 6.

One might view the ISO problem as just a document's failure to make clear that which cryptographers know quite well. We see it differently—as symptomatic of an overreaching understanding of BN. For years we have observed, in papers and talks, that people say, and believe, that "Encrypt-then-MAC works well, while MAC-then-Encrypt does not." But this claim should be understood as a specific fact about pE + MAC $\to$ pAE conversion. Viewed as a general, definitionally-robust statement about AE, the claim is without foundation.

A modern view of AE should entail a multiplicity of starting points and ending points. Yet not all starting points, or ending points, are equal. The ISO 19772 attack suggests that ivE makes a good starting point for GC; after all, the aim of GC is to support *generic* use of off-the-shelf primitives, and ivE nicely formalizes what is found on that shelf. Similarly, the nAE goal has proven to be the desired-in-practice ending point. We thus explore ivE + MAC $\to$ nAE conversion. We start off by assuming that the MAC can authenticate tuples of strings (a vecMAC). We then consider a universe of 160 candidate schemes, the A-schemes. Eight of these are *favored*: they are always secure when their underlying primitives are sound, and with good bounds. See Figure 2. One A-scheme is *transitional*: it has an inferior established bound. Three A-schemes are *elusive*: for them, we have been unable to generically establish security or insecurity. The remaining 148 A-schemes are meaningless or wrong. Next we show how to realize any of the favored A-scheme using a conventional string-input MAC (a strMAC). The resulting B-schemes are shown in Figure 3.

Two of the schemes given are already known. Scheme A4 is SIV [21] (apart from the fact that the latter permits vector-valued AD, a natural extension that we ignore), while B1 is EAX [6] (or the generalization of it called EAX2). Our treatment places these modes within a generic-composition framework. In the process, the correctness proof of each mode is actually simplified.

To ensure that we did not overlook any correct schemes, we initially used a computer to identify those with trivial attacks. We were left to deal with the more modest number of remaining schemes. The computer-assisted work was eventually rendered unnecessary by conventional proofs.

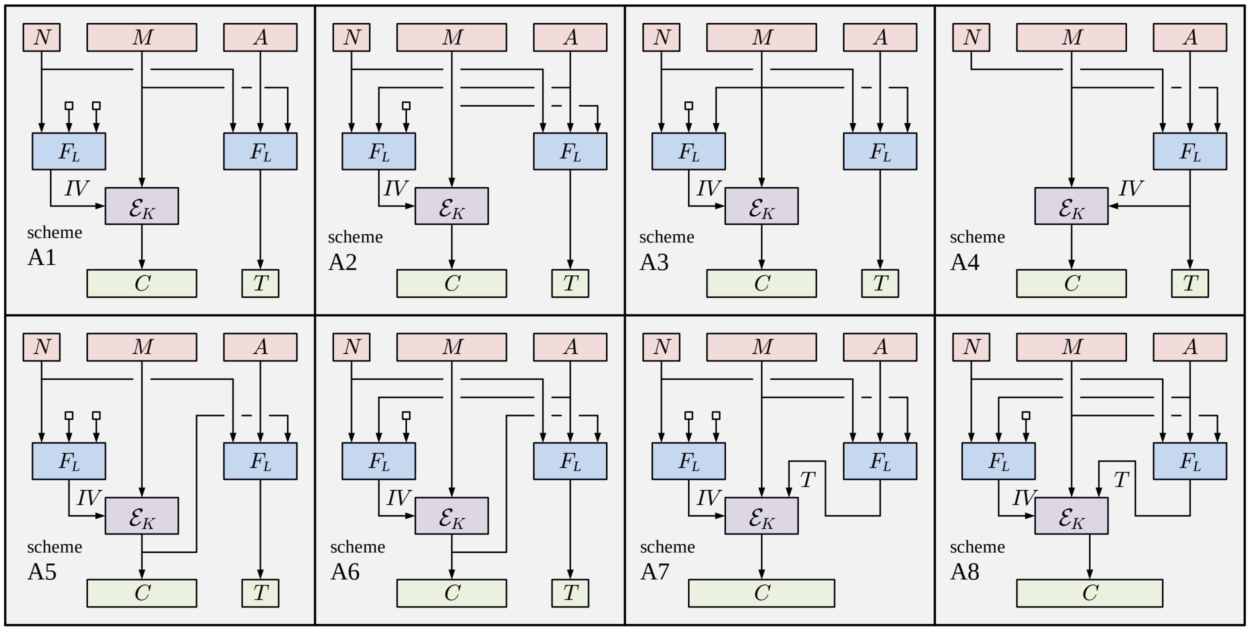

Fig. 2. The eight "favored" A-schemes. These convert an ivE scheme $\mathcal{E}$ and a vecMAC F into an nAE scheme. The IV is $F_L(N\,[,A]\,[,M])$ and the tag T is either $T = F_L(N,A,M)$ or $T = F_L(N,A,C)$. For this diagram we assume $F^{\text{iv}} = F^{\text{tag}} = F$.

We also look at the construction of nAE schemes from an nE scheme and a MAC [19, 20]. While nE schemes are not what practice directly provides—no more than pE schemes are—they are trivial to construct from an ivE scheme, and they mesh well with the nAE target. For this nE + MAC $\to$ nAE problem we identify 20 candidate schemes, which we call N-schemes. Three of them turn out to be secure, all with tight bounds. The security of one scheme we cannot resolve. The other 16 N-schemes are insecure.

Tidy encryption. Our formalization of ivE, nE, and nAE schemes includes a syntactic requirement, *tidiness*, that, when combined with the usual correctness requirement, demands that encryption and decryption be inverses of each other. (For an ivE scheme, correctness says that $\mathcal{E}_K(IV, M) = C \neq \bot$ implies that $\mathcal{D}_K(IV, C) = M$, while tidiness says that $\mathcal{D}_K(IV, C) = M \neq \bot$ implies that $\mathcal{E}_K(IV, M) = C$.) In the context of deterministic symmetric encryption, we regard *sloppy* schemes—those that are not tidy—as perilous in practice, and needlessly degenerate. Tidiness, we feel, is what one should demand.

Were sloppy nE and ivE schemes allowed, the generic composition story would shift again: only schemes A5 and A6, B5 and B6, and N2 would be generically secure. The sensitivity of GC to the sloppy/tidy distinction is another manifestation of the sensitivity of GC results to definitional choices.

A preemptive warning against misinterpretation. A body of results (eg, [8, 22]) have shown traditional MAC-then-Encrypt (MtE) schemes to be difficult to use properly in practice. Although some of our secure schemes can be viewed as being in the style of MtE, the results of this paper **should not** be interpreted as providing blanket support for MtE schemes. We urge extreme caution when applying *any* generic composition result from the literature, as implementers

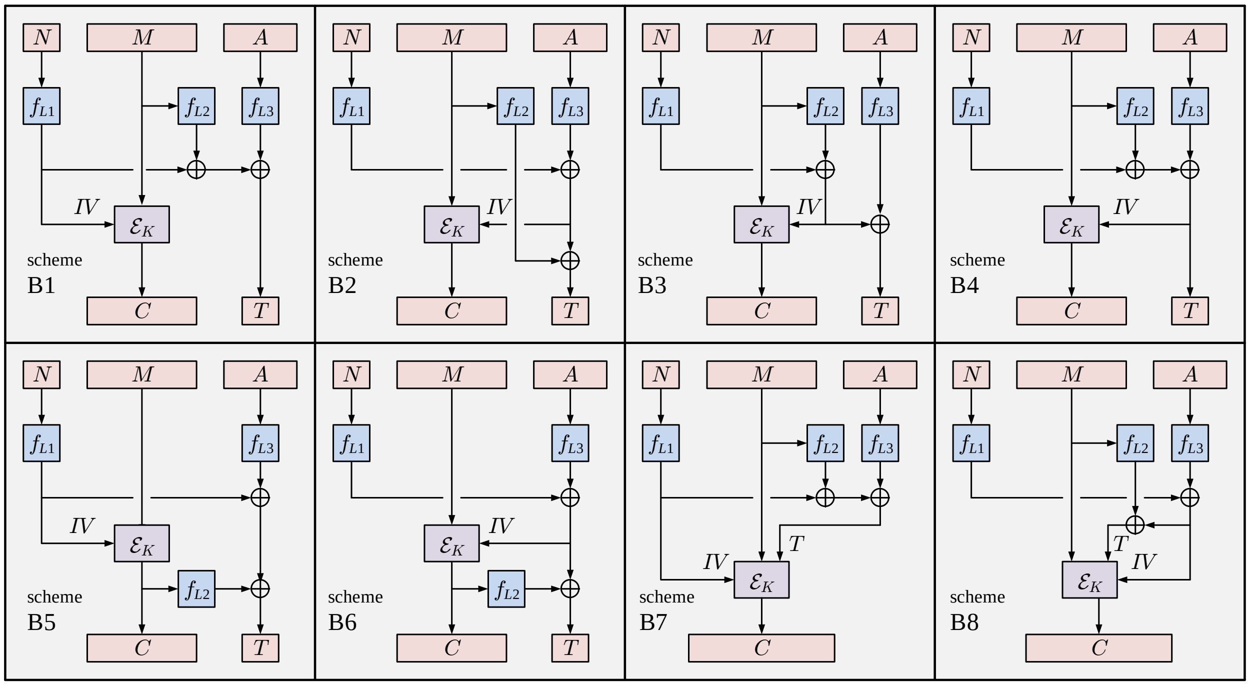

Fig. 3. The eight B-schemes corresponding to the favored A-schemes. Each converts an ivE scheme $\mathcal{E}$ and a strMAC f to an nAE scheme. The methods instantiate the vecMAC with a strMAC using the "XOR3" construction.

and standardizing bodies must insure that the underlying encryption and MAC primitives are of the type assumed by the result, and that they are composed in exactly the way the security result demands. Experience has demonstrated this area to be fraught with instantiation and usage difficulties.

Relatedly, we point out that our AE notions of security follow tradition in assuming that decryption failures return a *single kind of error message*, regardless of the cause. Hence implementations of our GC methods should insure, to the maximum extent possible, that this requirement is met.

Final introductory remarks. One might interpret our results as saying that the conventional wisdom—that Encrypt-then-MAC is the only safe GC method—is wrong, an artifact of early work having admitted sloppy schemes and considered only pE+MAC $\rightarrow$ pAE conversion. An alternative interpretation is that the conventional wisdom is essentially right, that Encrypt-then-MAC *is* the only safe GC method, for it works across multiple definitional settings, whereas the story becomes nuanced for other GC schemes.

Nothing in this paper should be understood as suggesting that there is anything wrong with BN. If that paper has been misconstrued, it was not for a lack of clarity. Our definitions and results are complementary.

We recently received a note from Bellare and Tackmann [7] pointing out that for the original nE definition of Rogaway [19], neither Encrypt-and-MAC nor MAC-then-Encrypt work for nE + MAC $\rightarrow$ nAE conversion, contradicting a (therefore buggy) theorem statement [19, Th. 7]. We had previously noticed the need to outlaw sloppy or length-increasing nE schemes to get these results to go through.

2 Definitions

This section provides key definitions. Some aspects are standard, but others (particularly tidiness, schemes recognizing their own domains, and identifying encryption schemes by their encryption algorithms) are not. We begin with some notation.

Notation. Strings are binary and finite. The length of x is $|x|$, $x \,\|\, y$ is the concatenation of x and y, and ε is the empty string. The set of all strings is $\{0,1\}^*$. If A is a probabilistic algorithm we can treat it as a distribution; if A is a distribution we can treat it as the set that is its support. If A is a distribution we write $a \twoheadleftarrow A$ for sampling from A and letting a be the result. If A is a finite set the uniform distribution is usually assumed.

We write $\Pr[S_1;\ S_2;\ \cdots;\ S_n\colon\ E]$ for the probability of event E after the experiment described by the preceding sequence of steps. When a function $F(x_1, \ldots, x_n)$ has multiple arguments, we may write them as subscripts, then superscripts, then parenthesized arguments. Thus $\mathcal{E}_K^{N,A}(M)$ means $\mathcal{E}(K, N, A, M)$. We use a superscript for functionality presented to an algorithm as an oracle; for example, $\mathcal{A}^F$ indicates that algorithm $\mathcal{A}$ has oracle access to F.

Kinds of encryption scheme. A scheme for **nonce-based AE** (nAE) is a triple $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$. The key space $\mathcal{K}$ is a finite nonempty set. Encryption algorithm $\mathcal{E}$ is deterministic and takes a four-tuple of strings K, N, A, M to a value $C \leftarrow \mathcal{E}_K^{N,A}(M)$ that is either a string or the symbol $\perp$ ("invalid"). We require the existence of sets $\mathcal{N}$, $\mathcal{A}$, and $\mathcal{M}$, the nonce space, associated-data space (AD space), and message space, such that $\mathcal{E}_K^{N,A}(M) \neq \perp$ iff $(K, N, A, M) \in \mathcal{K} \times \mathcal{N} \times \mathcal{A} \times \mathcal{M}$. We require that $\mathcal{M}$ contains two or more strings; that if $\mathcal{M}$ contains a string of length m it contains all strings of length m, and the same for $\mathcal{A}$; and that when $\mathcal{E}_K(N, A, M)$ is a string its length $\ell(|N|, |A|, |M|)$ depends only on $|N|$, $|A|$, and $|M|$. Decryption algorithm $\mathcal{D}$ is deterministic and takes a four-tuple of strings K, N, A, C to a value M that is either a string in $\mathcal{M}$ or the symbol $\perp$. We require that $\mathcal{E}$ and $\mathcal{D}$ be inverses of one another, implying:

(*Correctness*) if $\mathcal{E}_K^{N,A}(M) = C \neq \perp$ then $\mathcal{D}_K^{N,A}(C) = M$, and

(*Tidiness*) if $\mathcal{D}_K^{N,A}(C) = M \neq \perp$ then $\mathcal{E}_K^{N,A}(M) = C$.

Algorithm $\mathcal{D}$ is said to *reject* ciphertext C if $\mathcal{D}_K^{N,A}(C) = \perp$ and to *accept* it otherwise. Our security notion for a nAE scheme is given in Figure 4. The definition measures how well an adversary can distinguish an encryption-oracle / decryption-oracle pair from a corresponding pair of oracles that return random bits and $\perp$. Here and later, queries that would allow trivial wins are disallowed.

The syntax changes little when we are not expecting authenticity: schemes for **IV-based encryption** (ivE) and **nonce-based encryption** (nE) have the syntax above except for omitting all mention of AD. Security is specified in Figure 4. For ivE, the nonce N and nonce space $\mathcal{N}$ are renamed IV and $\mathcal{IV}$. With each query M the oracle selects a random IV and returns it alongside the ciphertext. For nE, the adversary provides a plaintext and a non-repeating nonce with each encryption query.

A scheme for **probabilistic encryption** (pE) or **probabilistic AE** (pAE) is a triple $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$. Key space $\mathcal{K}$ is a finite nonempty set. Encryption algorithm $\mathcal{E}$ is probabilistic and maps a pair of strings K, M to a value $C \twoheadleftarrow \mathcal{E}_K(M)$ that is either a string or the symbol $\perp$ ("invalid"). We require the existence of a set $\mathcal{M}$, the message space, such that $\mathcal{E}_K(M) \neq \perp$ iff $(K, M) \in \mathcal{K} \times \mathcal{M}$. We assume that $\mathcal{M}$ contains two or more strings and if $\mathcal{M}$ contains a string of length m then it contains all strings of length m. We demand that when $\mathcal{E}_K(M)$ is a string, its length $\ell(|M|)$ depends only on $|M|$. Decryption function $\mathcal{D}$ is deterministic and maps a pair of strings K, C to a value $M \leftarrow \mathcal{D}_K(C)$

that is either a string or the symbol $\perp$. We require *correctness*: if $\mathcal{E}_K(M) = C$ then $\mathcal{D}_K(C) = M$. Algorithm $\mathcal{D}$ *rejects* ciphertext C if $\mathcal{D}_K(C) = \perp$ and *accepts* it otherwise. Representative security definitions for pE and pAE schemes are given in Figure 4. For pE the adversary aims to distinguish an encryption oracle from an oracle that returns an appropriate number of random bits. For pAE the adversary also gets a decryption oracle or an oracle that always returns $\perp$.

Tidiness. In a pE or pAE scheme, what happens if the decryption algorithm $\mathcal{D}_K$ is fed an *illegitimate* ciphertext—a string C that is not the encryption of any string M under the key K? We didn't require $\mathcal{D}$ to reject, and perhaps it wouldn't make sense to, as a party has no realistic way to know, in general, if an alleged plaintext M for C would encrypt to it. But the situation is different for an ivE, nE, or nAE, as the decrypting party can easily check if a candidate plaintext M really does encrypt to a provided ciphertext C. And, in practice, this re-encryption never needs to be done: for real-world schemes, the natural decryption algorithm rejects illegitimate ciphertexts. Philosophically, once encryption and decryption become deterministic, one would expect them to be inverses of one another, as with a blockcipher.

An ivE scheme is *sloppy* if it satisfies everything but the tidiness condition. Might a "real world" ivE scheme be sloppy? The only case we know is when removal of padding is done wrong. Define $\mathcal{E}_K^{IV}(M) = \mathrm{CBC}_K^{IV}(M10^p)$, meaning CBC encryption over some n-bit blockcipher, with $p \geq 0$ the least number such that n divides $|M10^p|$. Let $\mathcal{D}_K^{IV}(C) = \perp$ if $|C|$ is not a positive multiple of n, and, otherwise, CBC-decrypt C to get M', strip away all trailing 0-bits, then strip any trailing 1-bit, then return what remains. Then any ciphertexts that CBC-decrypts to a string of zero-bits will give a plaintext of ε, which never encrypts to what we started from. So the method is sloppy. But it *should* be considered wrong: the intermediate plaintext M' was supposed to end in 10^p, for some $p \in [0..n-1]$, and if it did not, then $\perp$ should be returned. One is asking for trouble by silently accepting an improperly padded string.

Compact nomenclature. We formalized encryption schemes—all kinds—as tuples $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$. But tidiness means we don't need to specify decryption: given $\mathcal{E}$ one *must* have $\mathcal{D}_K(IV, C) = M$ if there is a (necessarily unique) $M \in \{0,1\}^*$ such that $\mathcal{E}_K(IV, M) = C$, and $\mathcal{D}_K(IV, C) = \perp$ otherwise. (For nonce-based schemes, rename IV as N; for nAE schemes, add in the the AD.) While there may still be reasons for writing down a decryption algorithm (eg, to demonstrate efficient computability), its not needed for well-definedness. We thus identify an ivE/nE/nAE scheme $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ by its encryption algorithm, writing $\mathcal{E}\colon \mathcal{K} \times \mathcal{IV} \times \mathcal{M} \to \{0,1\}^*$ for an ivE scheme, $\mathcal{E}\colon \mathcal{K} \times \mathcal{N} \times \mathcal{M} \to \{0,1\}^*$ for an nE scheme, and $\mathcal{E}\colon \mathcal{K} \times \mathcal{N} \times \mathcal{A} \times \mathcal{M} \to \{0,1\}^*$ for an nAE scheme. Remember, however, that the algorithm $\mathcal{E}$ can actually take in other strings outside the specified domain, retiring $\perp$ in such cases.

MACs. A message authentication code (MAC) is a deterministic algorithm F that takes in a key K and a value X and outputs either an n-bit string T or the symbol $\perp$. The *domain* of F is the set $\mathcal{X}$ such that $F_K(X) \neq \perp$ (we forbid this to depend on K). We write $F\colon \mathcal{K} \times \mathcal{X} \to \{0,1\}^n$ for a MAC with domain $\mathcal{X}$.

It is possible to be more general, considering probabilistic or stateful MACs, but neither primitive makes a suitable starting point when the goal is to create an nAE scheme. It is also possible to be less general; in particular, allowing the domain to be more than just strings is unusual. So is having the MAC recognize its own domain. Security of F is defined by $\mathbf{Adv}_F^{\mathrm{prf}}(\mathcal{A}) = \Pr[\mathcal{A}^F \Rightarrow 1] - \Pr[\mathcal{A}^\rho \Rightarrow 1]$. The game on the left selects $K \twoheadleftarrow \mathcal{K}$ and then provides the adversary an oracle for

$$\mathbf{Adv}_{\Pi}^{\mathrm{pE}}(\mathcal{A}) = \Pr\left[\mathcal{A}^{\mathcal{E}(\cdot)} \Rightarrow 1\right] - \Pr\left[\mathcal{A}^{\$(\cdot)} \Rightarrow 1\right]$$

where $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ is a pE scheme; $K \twoheadleftarrow \mathcal{K}$ at the beginning of each game; $\mathcal{E}(M)$ returns $C \twoheadleftarrow \mathcal{E}_K(M)$; and $\$(M)$ computes $C \twoheadleftarrow \mathcal{E}_K(M)$, returns $\perp$ if $C = \perp$, and otherwise returns $|C|$ random bits.

$$\mathbf{Adv}_{\Pi}^{\mathrm{pAE}}(\mathcal{A}) = \Pr\left[\mathcal{A}^{\mathcal{E}(\cdot),\, \mathcal{D}(\cdot)} \Rightarrow 1\right] - \Pr\left[\mathcal{A}^{\$(\cdot),\, \perp(\cdot)} \Rightarrow 1\right]$$

where $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ is a pAE scheme; $K \twoheadleftarrow \mathcal{K}$ at the beginning of each game; $\mathcal{E}(M)$ returns $C \twoheadleftarrow \mathcal{E}_K(M)$ and $\mathcal{D}(C)$ returns $\mathcal{D}_K(M)$; $\$(M)$ computes $C \twoheadleftarrow \mathcal{E}_K(M)$ and returns $\perp$ if $C = \perp$ and $|C|$ random bits otherwise; $\perp(M)$ returns $\perp$; and $\mathcal{A}$ may not make a decryption (=right) query C if C was returned by a prior encryption (=left) query.

$$\mathbf{Adv}_{\Pi}^{\mathrm{ivE}}(\mathcal{A}) = \Pr\left[\mathcal{A}^{\mathcal{E}(\cdot)} \Rightarrow 1\right] - \Pr\left[\mathcal{A}^{\$(\cdot)} \Rightarrow 1\right]$$

where $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ is an ivE scheme; $K \twoheadleftarrow \mathcal{K}$ at the beginning of each game; $\mathcal{E}(M)$ selects $IV \twoheadleftarrow \mathcal{IV}$ and returns $IV \parallel \mathcal{E}_K(IV, M)$; and $\$(M)$ selects $IV \twoheadleftarrow \mathcal{IV}$, computes $C = \mathcal{E}_K(IV, M)$, returns $\perp$ if $C = \perp$, and otherwise returns $|IV \parallel C|$ random bits.

$$\mathbf{Adv}_{\Pi}^{\mathrm{nE}}(\mathcal{A}) = \Pr\left[\mathcal{A}^{\mathcal{E}(\cdot,\cdot)} \Rightarrow 1\right] - \Pr\left[\mathcal{A}^{\$(\cdot,\cdot)} \Rightarrow 1\right]$$

where $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ is a nE scheme; $K \twoheadleftarrow \mathcal{K}$ at the beginning of each game; $\mathcal{E}(N, M)$ returns $\mathcal{E}_K(N, M)$; $\$(N, M)$ computes $C \leftarrow \mathcal{E}_K(N, M)$, returns $\perp$ if $C = \perp$, and otherwise returns $|C|$ random bits; and $\mathcal{A}$ may not repeat the first component of an oracle query.

$$\mathbf{Adv}_{\Pi}^{\mathrm{nAE}}(\mathcal{A}) = \Pr\left[\mathcal{A}^{\mathcal{E}(\cdot,\cdot,\cdot),\, \mathcal{D}(\cdot,\cdot,\cdot)} \Rightarrow 1\right] - \Pr\left[\mathcal{A}^{\$(\cdot,\cdot,\cdot),\, \perp(\cdot,\cdot,\cdot)} \Rightarrow 1\right]$$

where $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ is an nAE scheme; $K \twoheadleftarrow \mathcal{K}$ at the beginning of each game; $\mathcal{E}(N, A, M)$ returns $\mathcal{E}_K(N, A, M)$ and $\mathcal{D}(N, A, C)$ returns $\mathcal{D}_K(N, A, C)$; and $\$(N, A, M)$ computes $C \leftarrow \mathcal{E}_K(N, A, M)$, returns $\perp$ if $C = \perp$, and $|C|$ random bits otherwise, and $\perp(N, A, M)$ returns $\perp$; and $\mathcal{A}$ may not repeat the first component of an encryption (=left) query, nor make a decryption (=right) query (N, A, C) after C was obtained from a prior encryption (=left) query (N, A, M).

Fig. 4. Definitions for encryption: probabilistic encryption (pE), probabilistic authenticated encryption (pAE), iv-based encryption (ivE), nonce-based encryption (nE), and nonce-based AE (nAE). For consistency, we give ind\$-style notions throughout.

$F_K(\cdot)$. The game on the right selects a uniformly random function ρ from $\mathcal{X}$ to $\{0,1\}^n$ and provides the adversary an oracle for it. With either oracle, queries outside $\mathcal{X}$ return $\perp$. A *string-input* MAC (strMAC) (the conventional setting) has domain $\mathcal{X} \subseteq \{0,1\}^*$. A *vector-input* MAC (vecMAC) has a domain $\mathcal{X}$ with one or more component, and not necessarily strings.

Infectiousness of $\perp$. Encryption schemes and MACs return $\perp$ when applied to a point outside their domain. To specify algorithms without having tedious checks for this, we establish the convention that all functions return $\perp$ if any input is $\perp$. For example, if $T = \perp$ then $\mathcal{C} = C \parallel T$ is $\perp$; and if $IV = \perp$ then $C = \mathcal{E}_K(IV, M)$ is $\perp$.

3 AE from IV-Based Encryption and a Vector-Input MAC

We study a family of nAE constructions that combine an ivE encryption scheme and a MAC. The former is assumed to provide ind\$-style privacy when the IV is chosen uniformly and prepended to the ciphertext (ivE-security). The latter comes in two varieties, a vector-input MAC (vecMAC) and a string-input MAC (strMAC). This section assumes a vecMAC; the next section extends the treatment to a strMAC. Using a vecMAC provides a clean starting point for situations where one would like to authenticate a collection of typed values, like a nonce, AD, and plaintext. It is also a convenient waypoint for getting to ivE + strMAC $\to$ nAE.

Candidate schemes. We define a set of candidate schemes, the A-schemes, to make an nAE scheme out of an ivE scheme $\mathcal{E}\colon \mathcal{K}\times\{0,1\}^{\eta}\times\mathcal{M}\to\{0,1\}^{*}$, a vecMAC $F^{\mathrm{iv}}\colon \mathcal{L}\times\mathcal{X}^{\mathrm{iv}}\to\{0,1\}^{\eta}$, and a vecMAC $F^{\mathrm{tag}}\colon \mathcal{L}\times\mathcal{X}^{\mathrm{tag}}\to\{0,1\}^{\tau}$. Our constructions come in three types.

- **Type A$_1$ schemes**. The nAE scheme $\boldsymbol{\mathcal{E}} = \mathrm{A}_1.\mathrm{bbbbbb}[\mathcal{E}, F^{\mathrm{iv}}, F^{\mathrm{tag}}]$ defines $\boldsymbol{\mathcal{E}}_{K\,L}^{N,A}(M) = C \parallel T$ where
  $$IV = F_L^{\mathrm{iv}}(N\mid\sqcup,\ A\mid\sqcup,\ M\mid\sqcup)\ \text{ and }\ C=\mathcal{E}_K(IV,\ M)\ \text{ and }\ T=F_L^{\mathrm{tag}}(N\mid\sqcup,\ A\mid\sqcup,\ M\mid\sqcup).$$
  The notation $X \mid \sqcup$ means that the value is either the binary string X (the value is *present*) or the distinguished symbol $\sqcup$ (it is *absent*). The binary string bbbbbb $\in\{0,1\}^6$ specifies the chosen inputs to F^{iv} and F^{tag}, with 1 for present and 0 for absent, and ordered as above. For example, scheme $\mathrm{A}_1.100\,111[\mathcal{E}, F^{\mathrm{iv}}, F^{\mathrm{tag}}]$ sets $IV = F_L^{\mathrm{iv}}(N,\sqcup,\sqcup)$ and $T=F_L^{\mathrm{tag}}(N,A,M)$.
- **Type A$_2$ schemes**. The nAE scheme $\boldsymbol{\mathcal{E}} = \mathrm{A}_2.\mathrm{bbbbbb}[\mathcal{E}, F^{\mathrm{iv}}, F^{\mathrm{tag}}]$ defines $\boldsymbol{\mathcal{E}}_{K\,L}^{N,A}(M) = C \parallel T$ where
  $$IV = F_L^{\mathrm{iv}}(N\mid\sqcup,\ A\mid\sqcup,\ M\mid\sqcup)\ \text{ and }\ C=\mathcal{E}_K(IV,\ M)\ \text{ and }\ T=F_L^{\mathrm{tag}}(\ N\mid\sqcup,\ A\mid\sqcup,\ C).$$
  Notation is as above. In particular, bbbbbb remains a 6-bit string, but its final bit is fixed: it's always 1. (Nothing new would be included by allowing $\sqcup$ in place of C, since that's covered as a type A$_1$ scheme.)
- **Type A$_3$ schemes.** The nAE scheme $\boldsymbol{\mathcal{E}} = \mathrm{A}_3.\mathrm{bbbbbb}[\mathcal{E}, F^{\mathrm{iv}}, F^{\mathrm{tag}}]$ defines $\boldsymbol{\mathcal{E}}_{K\,L}^{N,A}(M) = C$ where
  $$IV = F_L^{\mathrm{iv}}(N\mid\sqcup,\ A\mid\sqcup,\ M\mid\sqcup)\ \text{ and }\ T=F_L^{\mathrm{tag}}(N\mid\sqcup,\ A\mid\sqcup,\ M\mid\sqcup)\ \text{ and }\ C=\mathcal{E}_K(IV, M\ \parallel\ T).$$

According to our conventions, the formulas above return $\boldsymbol{\mathcal{E}}_{K\,L}^{N,A}(M)=\perp$ if the calculation of IV, C, or T returns $\perp$. This happens when points are outside of the domain $\mathcal{E}$, F^{iv}, or F^{tag}.

Many of the "schemes" named above are not valid schemes: while there are a total of $2^6+2^5+2^6 = 160$ candidates, many will fail to satisfy the syntax of an nAE schemes. A candidate scheme might be invalid for all $(\mathcal{E}, F^{\mathrm{iv}}, F^{\mathrm{tag}})$, or it might be valid for some $(\mathcal{E}, F^{\mathrm{iv}}, F^{\mathrm{tag}})$ but not for others. We are only interested in candidate schemes $\boldsymbol{\mathcal{E}}$ with parameters $(\mathcal{E}, F^{\mathrm{iv}}, F^{\mathrm{tag}})$ that are *compatible*—ones where the specified composition does indeed satisfy the syntax of an nAE scheme. For example, with $\mathrm{A}_1.001\,111$ (where $IV = F_L^{\mathrm{iv}}(\sqcup,\sqcup,M)$ and $T=F_L^{\mathrm{tag}}(N,A,M)$) there will never be a way to decrypt. And even for a scheme like $\mathrm{A}_1.100\,111$ (where $IV=F_L^{\mathrm{iv}}(N,\sqcup,\sqcup)$ and $T=F_L^{\mathrm{tag}}(N,A,M)$), still we need for the domains to properly mesh. If they do not, the (non-)scheme is excluded from study.

Type A$_1$ and type A$_2$ schemes are *outer-tag* schemes, as T falls outside of what's encrypted by $\mathcal{E}$. Type A$_3$ schemes are *inner-tag* schemes, as T lies inside the scope of what's encrypted by $\mathcal{E}$. This distinction seems as compelling as the A$_1$, A$_2$, A$_3$ distinction that corresponds to E&M, EtM, and MtE style composition.

It is a *thesis* that our enumeration of A-schemes includes all natural ways to make an nAE scheme from an ivE scheme and a vecMAC. More specifically, the schemes are designed to exhaust all possibilities that employ one call to the ivE, two calls to the MAC, and one concatenation involving a MAC-produced tag.

Underlying PRF. It is unintuitive why, in the context of GC, we should use a common key L for components F^{iv} and F^{tag}. The choice enhances generality and uniformity of treatment: the two MACs have the *option* of employing non-overlapping portions of the key L (supporting key separation), but they are not obliged to do so (enabling a significant, additional scheme).

Yet common keying has drawbacks. When MACs F^0_L, F^1_L are queried on disjoint sets $\mathcal{X}_0, \mathcal{X}_1$ the pair need not resemble random functions ρ^0, ρ^1. To overcome this, retaining the generality and potential key-concision we seek, we assume that any $(F^{\text{iv}}, F^{\text{tag}})$ used to instantiate an A-scheme $\boldsymbol{\mathcal{E}} = \text{A}_i.\text{bbbbbb}[\mathcal{E}, F^{\text{iv}}, F^{\text{tag}}]$ can be derived from an underlying PRF $F\colon \mathcal{L} \times \mathcal{X} \to \{0,1\}^n$ by either

$$F^{\text{iv}}_L(\mathbf{x}) = F_L(\mathbf{x})[1\,..\,\eta] \;\text{ and }\; F^{\text{tag}}_L(\mathbf{x}) = F_L(\mathbf{x})[1\,..\,\tau], \;\text{ or} \tag{1}$$

$$F^{\text{iv}}_L(\mathbf{x}) = F_L(\mathsf{iv}, \mathbf{x})[1\,..\,\eta] \;\text{ and }\; F^{\text{tag}}_L(\mathbf{x}) = F_L(\mathsf{tag}, \mathbf{x})[1\,..\,\tau], \text{ for distinct constants } \mathsf{iv} \text{ and } \mathsf{tag}, \tag{2}$$

where $n \geq \max\{\eta, \tau\}$. In words, F^{iv} and F^{tag} must spring from an underlying PRF F, either with or without domain separation. The approach encompass all schemes that would arise by assuming independent keys for F^{iv} and F^{tag}, plus all schemes that arise by using a singly-keyed PRF for both of these MACs.

Summary of security results. We identify nine provably secure A-schemes, nicknamed A1–A9. See Figure 5. When one selects an ivE-scheme $\mathcal{E}$ and a MAC F that induces F^{iv} and F^{tag} so as to get a valid nAE scheme (which can always be done in these cases), these nine compositional methods are secure, assuming $\mathcal{E}$ is ivE-secure and F is PRF secure. The concrete bounds proven for A1–A8 are tight. The bound for A9 is inferior, due to the (somewhat curious) presence of ivE-advantage (i.e., privacy) term appearing in the authenticity bound. Additionally, the absence of the nonce N in the computation of F^{tag} prohibits its generic realization (by the construction we will give) from a conventional, string-input MAC. For these reasons we consider A1–A8 "better" than A9 and call them *favored*; A9 is termed *transitional.* The favored schemes are exactly those A-schemes for which the IV depends on (at least) the nonce N, while the tag T depends on everything: $T = F^{\text{tag}}_L(N, A, M)$ or $T = F_L(N, A, C)$.

Also shown in Figure 5 are three *elusive* schemes, A10, A11 and A12, whose status remains open. That they provide privacy (in the nE-sense) follows from the ivE-security of the underlying encryption scheme. But we have been unable to prove that these schemes provide authenticity under the same assumptions used for A1–A9. Nor have we been able to construct a counterexample to demonstrate that those assumptions do not suffice. (We have spent a considerable effort on both possibilities.) In Section E we discuss the technical difficulties encountered. We also prove there that A10, A11, and A12 do provide authenticity under an additional security assumption, what we call the *knowledge-of-tags* assumption. We do not know if this new assumption is implied by ivE-security.

All A-schemes other than A1–A12 are insecure, as we shall prove. We must do so in a systematic manner, of course, there being 148 such schemes.

For-free domain-separation. It's important to notice that for all secure and potentially secure A-schemes except A4, the *pattern* of arguments fed to F^{iv} and F^{tag} (ie, which arguments are present and which are absent) are distinct. In particular, the domain-of-application for these MACs are intrinsically separated: no vector $\mathbf{x}$ that might be fed to one MAC could ever be fed to the other. So, in all of these cases, there is no loss of generality to drop the domain-separation constants of equation (2). As for A4, the only natural way to achieve validity—for plaintexts to be recoverable from ciphertexts—is for $F^{\text{iv}}_K(\mathbf{x}) = F^{\text{tag}}_K(\mathbf{x}) = F_K(\mathbf{x})$. Our subsequent analysis assumes this for A4. In short, our security analysis establishes that there is no loss of generality to assume no domain separation, equation (1), for all secure A-schemes.

An	Scheme	IV	Tag	Sec	Comments	See
A1	A$_1$.100 111	$F_L^{\text{iv}}(N,\sqcup,\sqcup)$	$F_L^{\text{tag}}(N,A,M)$	yes	(Favored) Encrypt can compute C, T in parallel.	A.2
A2	A$_1$.110 111	$F_L^{\text{iv}}(N,A,\sqcup)$	$F_L^{\text{tag}}(N,A,M)$	yes	(Favored) Encrypt can compute C, T in parallel.	A.2
A3	A$_1$.101 111	$F_L^{\text{iv}}(N,\sqcup,M)$	$F_L^{\text{tag}}(N,A,M)$	yes	(Favored) Assume IV recoverable. Untruncatable.	A.2
A4	A$_1$.111 111	$F_L^{\text{iv}}(N,A,M)$	$F_L^{\text{tag}}(N,A,M)$	yes	(Favored) Assume $F^{\text{iv}} = F^{\text{tag}}$. Untruncatable. Nonce-reuse secure.	A.3
A5	A$_2$.100 111	$F_L^{\text{iv}}(N,\sqcup,\sqcup)$	$F_L^{\text{tag}}(N,A,C)$	yes	(Favored) Decrypt can validate T first, compute M, T in parallel.	A.2
A6	A$_2$.110 111	$F_L^{\text{iv}}(N,A,\sqcup)$	$F_L^{\text{tag}}(N,A,C)$	yes	(Favored) Decrypt can validate T first, compute M, T in parallel.	A.2
A7	A$_3$.100 111	$F_L^{\text{iv}}(N,\sqcup,\sqcup)$	$F_L^{\text{tag}}(N,A,M)$	yes	(Favored) Untruncatable.	A.2
A8	A$_3$.110 111	$F_L^{\text{iv}}(N,A,\sqcup)$	$F_L^{\text{tag}}(N,A,M)$	yes	(Favored) Untruncatable.	A.2
A9	A$_3$.110 101	$F_L^{\text{iv}}(N,A,\sqcup)$	$F_L^{\text{tag}}(N,\sqcup,M)$	yes	(Transitional) Weaker bound. Untruncatable.	A.4
A10	A$_3$.110 011	$F_L^{\text{iv}}(N,A,\sqcup)$	$F_L^{\text{tag}}(\sqcup,A,M)$	??	(Elusive) Security unresolved.	E
A11	A$_3$.110 001	$F_L^{\text{iv}}(N,A,\sqcup)$	$F_L^{\text{tag}}(\sqcup,\sqcup,M)$	??	(Elusive) Security unresolved.	E
A12	A$_3$.100 011	$F_L^{\text{iv}}(N,\sqcup,\sqcup)$	$F_L^{\text{tag}}(\sqcup,A,M)$	??	(Elusive) Security unresolved.	E
—	all others	—	—	no	Counterexamples given.	C

Fig. 5. Security of A-schemes: ivE+vecMAC $\to$nAE. The first column gives a nickname for the scheme. The next column gives the full name. The next two columns (formally redundant) serve as a reminder for how IV and T are determined. A "yes" in the "Sec" column means that we give a proof of security assuming ivE and PRF security for the primitives. A "no" means that we give a counterexample to such a proof existing. A "??" means that we have been unable to find a proof or counterexample. Comments include notes on security and efficiency. "Untruncatable" means that the tag T cannot be truncated. The "See" column indicates the Section where a proof can be found. Favored schemes were earlier pictured in Figure 2.

Theorems. We are now ready to state our results about the security of the A-schemes. For the proofs of Theorems 1 and 2, see Appendices A and C, respectively. The characterization leaves a small "hole" (schemes A10, A11, and A12); see Appendix E for discussion and results about those three schemes. For compactness, our theorem statements are somewhat qualitative. But the proofs give a quantitative analysis of the reductions and concrete bounds. For these details, see Figure 9.

Theorem 1 (Security of A1–A9). Fix a compositional method A$n \in \{\text{A1},\ldots,\text{A9}\}$ and let $\mathcal{E}\colon \mathcal{K}\times\{0,1\}^\eta\times\mathcal{M}\to\{0,1\}^*$ be an ivE-scheme. Fix integers $1\le\eta,\tau\le r$ and let $F\colon\mathcal{L}\times\mathcal{X}\to\{0,1\}^r$ be a vecMAC from which $F^{\text{iv}}\colon\mathcal{L}\times\mathcal{X}^{\text{iv}}\to\{0,1\}^\eta$ and $F^{\text{tag}}\colon\mathcal{L}\times\mathcal{X}^{\text{tag}}\to\{0,1\}^\tau$ are derived. Let the resulting nAE-scheme be denoted $\boldsymbol{\mathcal{E}} = \text{A}n[\mathcal{E},F^{\text{iv}},F^{\text{tag}}]$. Then there are blackbox reductions, explicitly given and analyzed in the proof of this theorem, that transform an adversary breaking the nAE-security of $\boldsymbol{\mathcal{E}}$ into adversaries breaking the ivE-security of $\mathcal{E}$, the PRF-security of F^{iv}, and the PRF-security of F^{tag}. For schemes A1–A8, the reductions are tight.

Theorem 2 (Insecurity of A-schemes other than A1–A12). Fix an A-compositional method other than A1–A12 and integers $1\le\eta,\tau\le r$. Then there is an ivE-secure encryption scheme $\mathcal{E}\colon\mathcal{K}\times\{0,1\}^\eta\times\mathcal{M}\to\{0,1\}^*$ and a vecMAC $F^{\text{iv}}\colon\mathcal{L}\times\mathcal{X}^{\text{iv}}\to\{0,1\}^\eta$ and $F^{\text{tag}}\colon\mathcal{L}\times\mathcal{X}^{\text{tag}}\to\{0,1\}^\tau$, derived from a a PRF-secure $F\colon\mathcal{L}\times\mathcal{X}\to\{0,1\}^r$, such that the resulting nAE-scheme is completely insecure. The claim holds under standard, scheme-dependent cryptographic assumptions stated in the proof.

Elusive schemes. It may seem surprising that the security status of schemes A10, A11, and A12 remains open. Indeed we initially thought that these schemes would admit (more-or-less) straightforward proofs or counterexamples, like other GC schemes. Let us give some intuition for some difficulties encountered.

Notice that in A10–A12, unlike A1–A9, the nonce is not an input to *both* F^{iv} and F^{tag}. For privacy, this causes no problems. But suppose an authenticity-attacking A11-adversary makes encryption queries (N_1, A_1, M_1) and (N_2, A_2, M_2), where $M_1 \neq M_2$, and then prepares a ciphertext query (N, A, C) with the following properties. First, $(N, A) = (N_1, A_1)$, which implies that the IV used for decrypting C is $IV_1 = F_L^{\text{iv}}(N_1, A_1)$. Second, C decrypts to $M_2 \,\|\, T_2$, where $T_2 = F_L^{\text{tag}}(M_2)$. For a tidy ivE-scheme $\mathcal{E}$, this means that $C = \mathcal{E}_K(IV_1, M_2 \,\|\, T_2)$. Since both IV_i and T_j were already computed, one cannot appeal to the PRF-security of F^{iv} or F^{tag}. Moreover, assuming that $\mathcal{E}$ is a secure ivE-scheme lets us conclude nothing about the (computational) randomness of $\mathcal{E}_K(IV_1, M_2 \,\|\, T_2)$ after having observed $C_1 = \mathcal{E}_K(IV_1, M_1 \,\|\, T_1)$. Intuitively, it should be hard for an adversary to create such an (N, A, C), in particular since the tags T_1 and T_2 should be hidden. Yet it is not at all obvious how to argue this case away. Similar situations can arise in A10 and A12.

Instead, one might try to create a counterexample, exploiting the characteristics of the vexing case above. (If sloppy $\mathcal{E}$ were allowed this would be easy.) But since IVs and tags should be random, any "useful" weaknesses built into $\mathcal{E}$ would likely need to be triggered by most IVs or tags. This would likely render $\mathcal{E}$ insecure, and useless as a counterexample. One might try to lightly modify F^{iv} and F^{tag} so that they enable $\mathcal{E}$ to selectively trigger weaknesses, thereby retaining ivE-security. But any reliable test that $\mathcal{E}$ could run on IV, T would likely result in an attack that breaks the PRF-security of F^{iv} or F^{tag}.

In Section E we give security proofs for A10, A11, and A12 by leveraging a new *knowledge-of-tags* assumption.

4 AE from IV-Based Encryption and a String-Input MAC

We turn our attention to achieving nAE from an ivE scheme and a conventional, string-input MAC. In place of our vector-input MAC we will call a string-input MAC multiple times, xoring the results.

There are two basic approaches to this enterprise. The first is to mimic the process already carried out in Section 3. One begins by identifying all candidate "B-schemes" *de novo*: methods that combine one call to an ivE scheme and three calls to a MAC algorithm, one for each of N, A, and either M or $C = \mathcal{E}_K(IV, M)$. The generated ciphertext is either $C \,\|\, T$ (outer-MAC schemes) or $C = \mathcal{E}_K(IV, M \,\|\, T)$ (inner-MAC schemes), where T is the xor of computed MAC values. Each MAC is computed using a different key, one of $L1$, $L2$, or $L3$. For each candidate scheme, one seeks either a proof of security (under the ivE and PRF assumptions) or a counter-example. Carrying out this treatment leads to a taxonomy paralleling that discovered for A-schemes.

A second approach is to leverage our ivE + vecMAC results, instantiating the secure schemes using a strMAC. On the downside, this does not give rise to a secure/insecure classification of all schemes cut from a common cloth. On the upside, it is simpler, and with it we identify a set of schemes desirable for a high-level reason: an abstraction boundary that lets us cleanly understand *why* security holds. Namely, it holds because the subject scheme is an instantiation of a scheme already known to be secure.

In the rest of this section, we follow the second approach, identifying nine secure ivE+strMAC $\rightarrow$ nAE schemes corresponding to A1–A9 (eight of them preferred, owing to the better bound). Scheme A10, by nature of its structure, does not admit the same generic strMAC instantiation that suffices for A1–A9. We drop it from consideration.

From strMAC to vecMAC. We recall that schemes A1–A9 can be regarded as depending on an ivE scheme $\mathcal{E}$ and a vecMAC $F\colon \mathcal{L} \times (\mathcal{N} \times (\mathcal{A} \cup \{\sqcup\}) \times (\mathcal{M} \cup \{\sqcup\})) \to \{0,1\}^r$ from which functions F^{iv} and F^{tag} are defined. Here, we give a method to transform a strMAC $f\colon \mathcal{L} \times \mathcal{X} \to \{0,1\}^r$ into a vecMAC $F\colon \mathcal{L}^3 \times (\mathcal{X} \times (\mathcal{X} \cup \{\sqcup\}) \times (\mathcal{X} \cup \{\sqcup\})) \to \{0,1\}^r$, in order to instantiate A1–A9. We do this via the *three-xor construction*, defined by

$$F_{L1,L2,L3}(N,A,M) = f'_{L1}(N) \oplus f'_{L2}(A) \oplus f'_{L3}(M) \quad \text{where} \quad f'_L(X) = \begin{cases} f_L(X) & \text{if } X \in \{0,1\}^*, \text{ and} \\ 0^n & \text{if } X = \sqcup \end{cases} \tag{3}$$

We write the construction $F = \text{XOR3}[f]$. Now the three-xor construction certainly does not work, *in general*, to transform a PRF with domain $\mathcal{X}$ to one with domain $\mathcal{X} \times (\mathcal{X} \cup \{\sqcup\}) \times (\mathcal{X} \cup \{\sqcup\})$; for example, an adversary that obtains, by queries, $Y_0 = F_{L1,L2,L3}(N, \sqcup, \sqcup)$ and $Y_1 = F_{L1,L2,L3}(N, \sqcup, M)$ and $Y_2 = F_{L1,L2,L3}(N, A, \sqcup)$ and $Y_3 = F_{L1,L2,L3}(N, A, M)$ can trivially distinguish if F is given by the xor-construction or is uniform: in the former case, $Y_3 = Y_0 \oplus Y_1 \oplus Y_2$. All the same, that the xor construction works well in the context of realizing any of schemes A1–A9.

For $k \geq 1$ a number, define a sequence of queries $(N_1, \cdots), \ldots, (N_q, \cdots)$ as *at-most-k-repeating* if no value N occurs as a first query coordinate more than k times. An adversary *at-most-k-repeats* if the sequence of queries it asks is at-most-k-repeating, regardless of query responses.

Our observation is that, if f is a good PRF, then XOR3$[f]$ is a good PRF when restricted to at-most-2-repeats adversaries. We omit the proof.

Lemma 1 (XOR3 construction). Fix $r \geq 1$, let $f\colon \mathcal{L} \times \mathcal{X} \to \{0,1\}^r$, and let $F = \text{XOR3}[f]$. There is an explicitly given blackbox reduction $\mathcal{B}$ with the following property: for any at-most-2-repeats adversary $\mathcal{A}_F$ there is an adversary $\mathcal{A}_f = \mathcal{B}(\mathcal{A}_F)$ such that $\mathbf{Adv}_f^{\text{prf}}(\mathcal{A}_f) \geq \mathbf{Adv}_F^{\text{prf}}(\mathcal{A}_F)$. Adversary $\mathcal{A}_f$ makes at most three times the number of queries as $\mathcal{A}_F$, the total length μ of those queries is unchanged, and the running time of $\mathcal{A}_f$ is essentially unchanged as well.

To apply Lemma 1 we use the characterization of nAE security that allows the adversary only a single decryption query [21]. This notion is equivalent to our nAE notion of security (which gives the adversary an arbitrary number of decryption queries) apart from a multiplicative degradation in the security bound by a factor of q_{d}, the number of decryption queries. But for the 1-decryption game, the sequence of adversarial queries is at-most-2-repeating (no repetitions among encryption queries; then a single nonce-repetition for the decryption query). As a result, there is no significant loss in using XOR3$[f]$ to instantiate a vecMAC F

We conclude that the underlying MAC F of all favored A-scheme, and also A9, can be realized by the XOR3 construction. There is a quantitative loss of q_{d}, which is due to the "weaker" definition for nAE security; we have not determined if this loss is artifactual or necessary. In Figure 3 we draw the eight B schemes obtained by applying the XOR3 construction to the corresponding A-schemes. Methods B1 and B4 essentially coincide with EAX and SIV [6, 21], neither of which was viewed as an instance of a framework like that described here.

Collapsing the PRF keys. For simplicity, we defined the XOR3 construction as using three different keys. But of course we can realize f_{L1}, f_{L2}, f_{L3} by, for example, $f_{L1}(X) = f_L(\mathsf{c1} \,\|\, X)$, $f_{L2}(X) = f_L(\mathsf{c2} \,\|\, X)$, and $f_{L3}(X) = f_L(\mathsf{c3} \,\|\, X)$, for distinct, equal-length constants $\mathsf{c1}, \mathsf{c2}, \mathsf{c3}$.

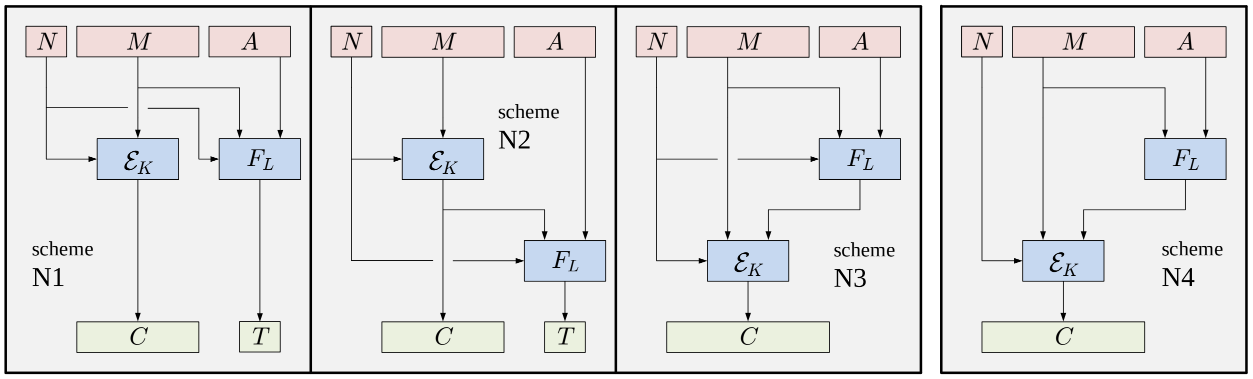

Fig. 6. Three correct N-schemes (left) and an elusive one (right). The methods achieve nE+vecMAC $\to$ nAE conversion. Application of the XOR3 construction to N1, N2, and N3 will result in three corresponding schemes that achieve nE + strMAC $\to$ nAE conversion. A second application of the XOR3 construction will recover the ivE + strMAC $\to$ nAE constructions B1, B5, and B7.

5 AE from Nonce-Based Encryption and a MAC

We study nAE constructions obtained by generically combining an nE encryption scheme and a MAC. The problem was previously investigated Rogaway [19]. (We compare our results at the end of this section.) The nE scheme from which we start is assumed to provide ind\$-style privacy when the nonce is never repeated (nE-security), while the MAC can be either a strMAC or a vecMAC. We focus on the latter, as the XOR3 construction can again be used to convert to to a secure nE + strMAC scheme. Our treatment follows, but abbreviates, that of Section 3, as the current setting is substantially simpler.

Candidate schemes. We define schemes, the N-schemes, to make an nAE scheme from an nE scheme $\mathcal{E}\colon \mathcal{K}\times\{0,1\}^\eta\times\mathcal{M}\to\{0,1\}^*$ and a vecMAC $F\colon \mathcal{L}\times\mathcal{X}\to\{0,1\}^\tau$. Our constructions come in three types.

- **Type N$_1$ schemes**. The nAE scheme $\boldsymbol{\mathcal{E}} = \mathrm{N}_1.\mathrm{bbb}[\mathcal{E}, F]$ defines $\boldsymbol{\mathcal{E}}^{N,A}_{K\,L}(M) \;=\; C \;\|\; T$ where
$$C \;=\; \mathcal{E}_K(N,\; M) \;\text{ and }\; T \;=\; F_L(N \mid \sqcup,\; A \mid \sqcup,\; M \mid \sqcup).$$
- **Type N$_2$ schemes**. The nAE scheme $\boldsymbol{\mathcal{E}} = \mathrm{N}_2.\mathrm{bbb}[\mathcal{E}, F]$ defines $\boldsymbol{\mathcal{E}}^{N,A}_{K\,L}(M) \;=\; C \;\|\; T$ where
$$C \;=\; \mathcal{E}_K(N,\; M) \;\text{ and }\; T \;=\; F_L(\, N \mid \sqcup,\; A \mid \sqcup,\; C).$$
We again take bbb $\in \{0,1\}^3$, but the third bit must be one.
- **Type N$_3$ schemes.** The nAE scheme $\boldsymbol{\mathcal{E}} = \mathrm{N}_3.\mathrm{bbb}[\mathcal{E}, F]$ defines $\boldsymbol{\mathcal{E}}^{N,A}_{K\,L}(M) \;=\; C$ where
$$T \;=\; F_L(N \mid \sqcup,\; A \mid \sqcup,\; M \mid \sqcup) \;\text{ and }\; C \;=\; \mathcal{E}_K(N, M \;\|\; T).$$

As before, the formulas return $\boldsymbol{\mathcal{E}}^{N,A}_{K\,L}(M) = \perp$ if the calculation of C or T returns $\perp$.

There are a total of $2^3 + 2^2 + 2^3 = 20$ candidate schemes, but many fail to satisfy the syntax of an nAE scheme. We are only interested in candidate methods that are valid nAE schemes.

Security results. We identify three provably secure schemes, nicknamed N1, N2, and N3. See Figure 6 and 7. The methods are secure when $\mathcal{E}$ is nE-secure and F is PRF-secure. For all three schemes, the concrete bounds are tight. Also shown in Figure 5 is a scheme N4 whose status remains open. Similar to the elusive A-schemes, N4 provides privacy (in the nE-sense), as follows from the

Nn	Scheme	Tag	Secure?	Comments	See §
N1	N$_1$.111	$F_L(N, A, M)$	yes	(Favored) Encrypt can compute C, T in parallel	B
N2	N$_2$.111	$F_L(N, A, M)$	yes	(Favored) Decrypt can validate T first, compute M, T in parallel	B
N3	N$_3$.111	$F_L(N, A, M)$	yes	(Favored) Untruncatable.	B
N4	N$_3$.011	$F_L(\sqcup, A, M)$	??	(Elusive) Security unresolved. Tag untruncatable.	—
—	all others	—	no	Counterexamples given.	D

Fig. 7. Security of N-schemes: nE+vecMAC $\rightarrow$ nAE.

nE-security of the underlying encryption scheme. But we have been unable to prove that N4 provides authenticity (under the same assumptions used for N1–N3); nor have we been able to construct a counterexample to demonstrate that the nE and PRF assumptions do not suffice. The technical difficulties are similar to those encountered in the attempts to deal with A11 and A12. As for N-schemes other than N1–N4, all 16 are insecure; we exhibit attacks in [18].

Theorems. We now state our results about the security of the N-schemes. For proofs, see Appendices B and D. The combination leaves a small "hole," which is scheme N4. For compactness, our theorem statements are again somewhat qualitative. But the proofs (Appendices B and D) are not. They provide explicit reductions and quantitative analyses. For concrete bounds, see Figure 9.

Theorem 3 (Security of N1–N3). Fix a compositional method N$n \in \{\text{N1}, \text{N2}, \text{N3}\}$ and integer $\tau \geq 1$. Fix an nE-scheme $\mathcal{E}\colon \mathcal{K} \times \{0,1\}^\eta \times \mathcal{M} \to \{0,1\}^*$ and a vecMAC $F\colon \mathcal{L} \times \mathcal{X} \to \{0,1\}^\tau$ that results in a valid nAE scheme $\boldsymbol{\mathcal{E}} = \text{N}n[\mathcal{E}, F]$. Then there are blackbox reductions, explicitly given and analyzed in the proof of this theorem, that transform an adversary breaking the nAE-security of $\boldsymbol{\mathcal{E}}$ to adversaries breaking the nE-security of $\mathcal{E}$ and the PRF-security of F. The reductions are tight.

A claim that N2 and N3 correctly accomplish nE + vecMAC $\rightarrow$ nAE conversion appears in earlier work by Rogaway [19, 20]. As pointed out by Bellare and Tackmann [7], the claim there was wrong for N3, as Rogaway's definitions had permitted sloppy schemes. This would make a counterexample for N3 (and also for N1) straightforward.

6 The ISO-Standard for Generic Composition

In this section we consider the Encrypt-then-MAC (EtM) mechanism of the ISO 19772 standard [15, Section 10]. We explore what went wrong, and why.

The problem. The EtM method of ISO 19772 (mechanism 5; henceforth isoEtM) combines a conventional encryption mode $\mathcal{E}$ and a MAC f.[4]For the former the standard allows CBC, CFB, OFB, or CTR—any ISO 10116 [14] scheme except ECB. For the MAC, f, the standard permits any of the algorithms of ISO 9797 [16]. These are variants of the CBC MAC. The latest edition of the standard names six CBC MAC variants, but the actual number is greater, as there are multiple possibilities for padding and key-separation.

The standard describes isoEtM encryption in just nine lines of text. After choosing an appropriate "starting variable" (SV) S for encryption mode $\mathcal{E}$, we're told to encrypt plaintext D to

[4] This section mostly follows naming conventions of the ISO standard, rather than the names used elsewhere in this paper.

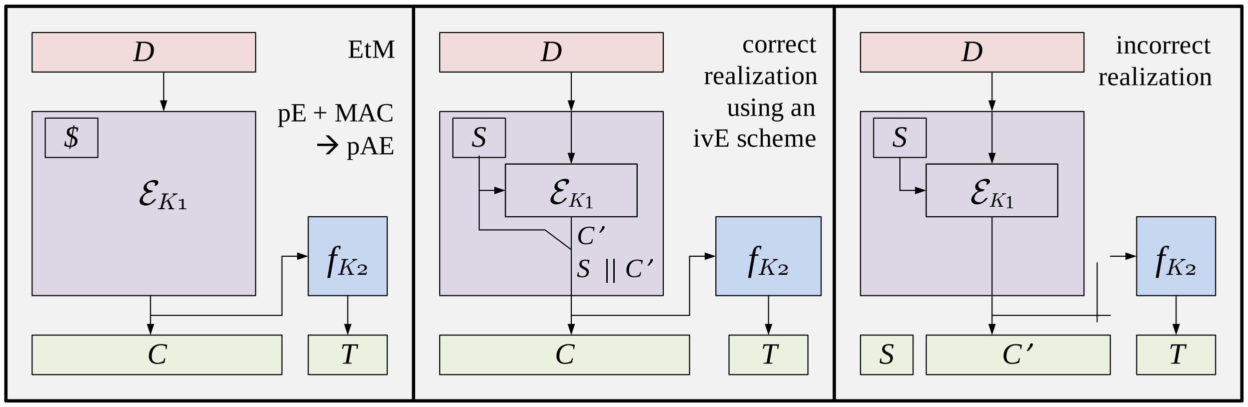

Fig. 8. Possible provenance of the ISO 19772 error. *Left*: The EtM method of BN, employing a probabilistic encryption algorithm $\mathcal{E}$ and a MAC f. The final ciphertext is $\mathcal{C} = C \,\|\, T$. *Middle*: A correct instantiation of EtM using an IV-based encryption scheme. With each encryption a random S is generated and embedded in C. The final ciphertext is $\mathcal{C} = C \,\|\, T$. *Right*: Mechanism 5 of ISO 19772. We can consider the final ciphertext as $\mathcal{C} = S \,\|\, C \,\|\, T$, but the string S is never MACed.

ciphertext $C = C' \,\|\, T$ by setting $C' = \mathcal{E}_{K_1}(D)$ and $T = f_{K_2}(C')$. In describing what "appropriate" means for S, the standard asserts that *[t]his variable shall be distinct for every message to be protected during the lifetime of a key, and must be made available to the recipient of the message* [15, p. 14]. It continues: *Further possible requirements for S are described in the appropriate clauses of ISO 10116.* The document levies no requirements on SV, but an annex says that a *randomly chosen statistically unique SV is recommended* [14, Annex B].

We aren't certain what this last phrase means, but suppose it to urge the use of uniformly random bits. But that possibility runs contrary to the requirement that SV not repeat. One is left to wonder if the SV is a nonce, a random value, or something else. But even if one insists that SV be uniformly random, still we have the biggest problem: ISO 10116 makes clear that the SV it is not a part of the ciphertext C' one gets from applying the encryption mode $\mathcal{E}$. The SV is separate from the ciphertext, communicated out-of-band. The result is that isoEtM never provides authenticity for SV, which leads to trivial attacks. See Figure 8. For example, let the adversary ask for the encryption of any message, obtaining a ciphertext $C = C' \,\|\, T$ and its associated SV S. Then a valid forgery is C itself, along with any SV S' other than S. Attacks like this break not only the AE property, but also weaker aims, like nonmalleability.

There are further problems with isoEtM. The standard asserts that *To prevent information leakage an integrity-protected secret SV is recommended.* The intent is baffling, as though the specified encryption modes are not self-contained primitives, but part of a larger cryptographic process. The AE scheme of the standard could potentially be such an enclosing process, but its SV is not integrity or privacy protected.

The correspondingly terse isoEtM decryption process partitions the ciphertext C into C' and T, outputs INVALID if $T \neq f_{K_2}(C')$, and outputs $\delta_{K_1}(C)$ otherwise, with δ the underlying mode's decryption algorithm. No mention is made of the SV. It is unclear what happens if $\delta_{K_1}(C)$ results in an error, a possibility implicit from the fact that padding is anticipated with ISO 10772 schemes yet considered considered out-of-scope.

Overall, it is unclear if isoEtM aims to provide pAE, nAE (without AD), or something else. But the omission of the SV from the scope of the MAC renders the method incorrect no matter what. There is no clear message space for the scheme, as padding is implicit and out of scope. It is unclear what one is supposed to do, on decryption, when padding problems arise. As for the MACs themselves, some ISO 9797 schemes are insecure when message lengths vary, a problem inherited by the enclosing AE scheme. There is no support for AD.

Diagnosis. ISO 19772 standardized five additional AE schemes, and we notice no problems with any of them. (A minor bug in the definition of GCM was pointed out by others, and is currently being corrected[17].) Why did the committee have bigger problems with (the conceptually simpler) GC?

When Bellare and Namprempre formalized Encrypt-then-MAC they assumed probabilistic encryption as the starting point. This is what any theory-trained cryptographer would have done at that time. But pE has remained a theorists' conceptualization: it is not an abstraction boundary widely understood by practitioners, realized by standards, embodied in APIs, or explained in popular books. Using this starting point within a standard is unlike building a scheme from a blockcipher, a primitive that *is* widely understood by practitioners, realized by standards, embodied in APIs, and explained in popular books. Given the difference between pE and actual, standardized encryption schemes, and given GC's sensitivity to definitional and algorithmic adjustments, it seems, in retrospect, a setting for which people are likely to err.

Acknowledgments. Thanks to Yusi (James) Zhang for identifying an error in a proof that formerly resulted in scheme A10 being classified as transitional instead of elusive. Thanks to the anonymous reviewers for their comments. Thanks to the NSF, who sponsored Rogaway's work under NSF grants CNS 0904380, CNS 1228828, and CNS 1314885, and Shrimpton's work under NSF grants CNS 0845610 and CNS 1319061. Much of Rogaway's work on this paper was done while he was visiting Thammasat University during the summer of 2013.

References

1. A. Alkassar, A. Geraldy, B. Pfitzmann, and A.-R. Sadeghi. Optimized self-synchronizing mode of operation. In M. Matsui, editor, *FSE 2001*, volume 2355 of *LNCS*, pages 78–91. Springer, Apr. 2001.
2. 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.
3. 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, Dec. 2000.
4. M. Bellare and C. Namprempre. Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. *Journal of Cryptology*, 21(4):469–491, Oct. 2008.
5. 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, Dec. 2000.
6. M. Bellare, P. Rogaway, and D. Wagner. The EAX mode of operation. In B. K. Roy and W. Meier, editors, *FSE 2004*, volume 3017 of *LNCS*, pages 389–407. Springer, Feb. 2004.
7. M. Bellare and B. Tackmann. Insecurity of MtE (and M&E) AEAD. Personal communications (unpublished note), July 2013.
8. B. Canvel, A. P. Hiltgen, S. Vaudenay, and M. Vuagnoux. Password interception in a SSL/TLS channel. In D. Boneh, editor, *CRYPTO 2003*, volume 2729 of *LNCS*, pages 583–599. Springer, Aug. 2003.
9. M. Dworkin. Recommendation for block cipher modes of operation: Methods and techniques. NIST Special Publication 800-38B, December 2001.
10. M. Dworkin. Recommendation for block cipher modes of operation: The CCM mode for authentication and confidentiality. NIST Special Publication 800-38C, May 2004.

11. M. Dworkin. Recommendation for block cipher modes of operation: Galois/counter mode (GCM) and GMAC. NIST Special Publication 800-38D, Nov 2007.
12. FIPS Publication 81. DES modes of operation. National Institute of Standards and Technology, U.S. Department of Commerce, Dec 1980.
13. S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270–299, 1984.
14. ISO/IEC 10116. Information technology — Security techniques — Modes of operation of an n-bit cipher. Third edition, 2006.
15. ISO/IEC 19772. Information technology — Security techniques — Authenticated encryption. First edition, 2009.
16. ISO/IEC 9797-1. Information technology — Security techniques — Message Authentication Codes (MACs) — Part 1: Mechanisms using a block cipher, 2011.
17. C. Mitchell. Personal communications, August 2011.
18. C. Namprempre, P. Rogaway, and T. Shrimpton. Reconsidering generic composition. Cryptology ePrint Archive, Report 2014/xxx, 2014. Full version of this paper.
19. P. Rogaway. Authenticated-encryption with associated-data. In V. Atluri, editor, *ACM CCS 02*, pages 98–107. ACM Press, Nov. 2002.
20. P. Rogaway. Nonce-based symmetric encryption. In B. K. Roy and W. Meier, editors, *FSE 2004*, volume 3017 of *LNCS*, pages 348–359. Springer, Feb. 2004.
21. 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, May / June 2006.
22. S. Vaudenay. Security flaws induced by CBC padding — applications to SSL, IPSEC, WTLS,... In L. R. Knudsen, editor, *EUROCRYPT 2002*, volume 2332 of *LNCS*, pages 534–545. Springer, Apr. / May 2002.

A Proofs of Security for A-Schemes

A.1 Proof Preliminaries

Resource conventions. Let G be an algorithm, possibly randomized and oracle-querying. An adversary $\mathcal{A}$ is an example that is both randomized and oracle-querying. Let $t_G(\ell)$ be the maximum amount of time to compute G on any sequence of inputs $X_1, X_2, \ldots, X_q$ of total bitlength at most ℓ. We write t_G for an overall maximum. If G makes blackbox calls to an algorithm $\mathcal{O}$, the time to compute $\mathcal{O}$ is charged to G.

When an algorithm makes an oracle query, we define the length of the query to be the sum of the bitlength of the query itself and the bitlength of the oracle response. The bitlength of tuples and non-string symbols is relative to some fixed, implicit encoding.

For the pE, ivE, nE and PRF-security notions notions we count the number of queries q made by the adversary $\mathcal{A}$ to its single oracle, these having a total length of μ. For the pAE and nAE notions we count the number of queries q_e make by $\mathcal{A}$ to its encryption oracle, and the number of queries q_d to its decryption oracle. The total length of these queries are μ_e and μ_d.

Auth security. Let $\mathcal{E}$ be an nAE scheme and $\mathcal{A}$ an adversary. We define the auth-advantage of $\mathcal{A}$ attacking $\mathcal{E}$ as $\mathbf{Adv}^{\mathrm{auth}}_{\mathcal{E}}(\mathcal{A}) = \Pr\left[\, K \twoheadleftarrow \mathcal{K} :\ \mathcal{A}^{\mathcal{E}_K, \mathcal{D}^*_K} \text{ gets its second oracle to return } 1 \,\right]$. Here, on input (N, A, M), oracle $\mathcal{E}_K$ returns $\mathcal{E}_K(N, A, M)$; while, on input (N, A, C), oracle $\mathcal{D}^*_K(N, A, C)$ returns 1 if $\mathcal{D}(K, N, A, C) \neq \perp$, and 0 otherwise. The advantage measure $\mathbf{Adv}^{\mathrm{auth1}}_{\mathcal{E}}(\mathcal{A})$ is the same, with the restriction that $\mathcal{A}$ makes a single decryption query (without loss of generality, its final query).

We make frequent use of the following lemma. Informally, it says that analyzing the security of an nAE scheme with respect to adversaries asking a single decryption query suffices to conclude security against adversaries asking multiple queries. The proof is a straightforward hybrid argument (over the decryption queries), so we omit it.

Scheme $\boldsymbol{\mathcal{E}}$	$\mathbf{Adv}^{\mathrm{nAE}}_{\boldsymbol{\mathcal{E}}}(\mathcal{A}) \le \mathbf{Adv}^{\mathrm{prf}}_{F}(\mathcal{B}(\mathcal{A})) + \mathbf{Adv}^{\mathrm{ivE}}_{\mathcal{E}}(\mathcal{D}_1(\mathcal{A})) + \dots$	$\mathcal{D}_2$ Resources
A1–A3, A5–A8	$q_d/2^\tau$	—
A4	$q_d/2^n$	—
A9	$q_d\left(\mathbf{Adv}^{\mathrm{ivE}}_{\mathcal{E}}(\mathcal{D}_2(\mathcal{A})) + 2/2^\tau\right)$	$t' \le t_A + t_{\mathcal{E}}(2\mu_e) + c(\mu + nq_e)$, $q' = q_e + 1$, $\mu' < 2\mu_e$
N1–N3	$q_d/2^\tau$	—

Fig. 9. Summary of concrete bounds for A1–A9, and N1–N3. For A1–A9, each nAE-scheme $\boldsymbol{\mathcal{E}}$ is a composition of ivE-scheme $\mathcal{E}$ and vecMAC F (from which F^{iv} and F^{tag} are derived). For N1–N3, each nAE-scheme $\boldsymbol{\mathcal{E}}$ is a composition of nE-scheme $\mathcal{E}$ and vecMAC F (from which F^{iv} and F^{tag} are derived). Reductions $\mathcal{B}$ and $\mathcal{D}_1$ are described in Lemma 3 (for A1–A9) and Lemma 4 (for N1–N3); these are common to the bounds of all schemes. The middle column gives the additional terms that are unique to the indicated schemes, and the final column gives the resources used by additional reductions. Resources are always with respect to the $t_{\mathcal{A}}, q_e, q_d, \mu_e, \mu_d$ associated to $\mathcal{A}$. The value c is a constant implicit in proofs.

Lemma 2. Let $\mathcal{E}$ be an nAE scheme. Let $\mathcal{A}$ be an adversary (for attacking $\mathcal{E}$), asking q_e queries to its encryption oracle of total length μ_e, and q_d queries to its decryption oracle of total length μ_d. Let t_A be the running time of A given these queries. Then there is an (explicitly known) blackbox reduction $\mathcal{B}$ such that $\mathbf{Adv}^{\mathrm{auth}}_{\Pi}(\mathcal{A}) \le q_d \cdot \mathbf{Adv}^{\mathrm{auth1}}_{\Pi}(\mathcal{B})$. Reduction $\mathcal{B}$ has time complexity $t_A + t_{\mathcal{E}}(\mu_e) + c(\mu_d)$ for a constant c; it asks at most q_e queries to its encryption oracle, these having total length μ_e, and makes one query to its decryption oracle, having total length at most μ_d.

The proofs for A1–A9 all share a common opening, which we capture in the following lemma. It allows us to move directly to the most interesting portions of those proofs.

Lemma 3. Fix a compositional method An $\in \{\mathrm{A1}, \dots, \mathrm{A9}\}$, and let $\mathcal{E}\colon \mathcal{K} \times \{0,1\}^\eta \times \mathcal{M} \to \{0,1\}^*$ be an ivE-scheme. Fix integers $0 < \eta, \tau \le r$, and let $F\colon \mathcal{L} \times \mathcal{X} \to \{0,1\}^r$ be a vecMAC. From F, derive $F^{\mathrm{iv}}\colon \mathcal{L} \times \mathcal{X}^{\mathrm{iv}} \to \{0,1\}^\eta$ and $F^{\mathrm{tag}}\colon \mathcal{L} \times \mathcal{X}^{\mathrm{tag}} \to \{0,1\}^\tau$ according to our conventions, so that the resulting nAE-scheme $\boldsymbol{\mathcal{E}} = \mathrm{An}[\mathcal{E}, F^{\mathrm{iv}}, F^{\mathrm{tag}}]$ is syntactically valid. Let $\mathcal{A}$ be an nAE-adversary for $\boldsymbol{\mathcal{E}}$. Then there are blackbox reductions $\mathcal{B}$ and $\mathcal{D}_1$, explicitly given and analyzed in the proof of this theorem, such that

$$\mathbf{Adv}^{\mathrm{nAE}}_{\boldsymbol{\mathcal{E}}}(\mathcal{A}) \le \mathbf{Adv}^{\mathrm{prf}}_{F}(\mathcal{B}(\mathcal{A})) + \mathbf{Adv}^{\mathrm{auth}}_{\tilde{\boldsymbol{\mathcal{E}}}}(\mathcal{A}) + \mathbf{Adv}^{\mathrm{ivE}}_{\mathcal{E}}(\mathcal{D}_1(\mathcal{A}))$$

where $\tilde{\boldsymbol{\mathcal{E}}}$ is shorthand for the composition $\mathrm{An}[\mathcal{E}, \rho_{\mathrm{iv}}, \rho_{\mathrm{tag}}]$ using random functions ρ_{iv} and ρ_{tag} in place of F^{iv} and F^{tag} (respectively). Say $\mathcal{A}$ asks q_e queries to its encryption oracle, and q_d queries to its decryption oracle, with $q = q_e + q_d$. The encryption queries have total length μ_e, and the decryption queries have total length μ_d, with $\mu = \mu_e + \mu_d$. Let t_A be the running time of A, given this total number and length of queries. Then reduction $\mathcal{B}$ has running time at most $t_A + 2t_F(\mu) + t_{\mathcal{E}}(\mu)$, asks at most $2(q_e + q_d)$ queries to its oracle, with total length at most μ. Reduction $\mathcal{D}_1$ has running time $t_A + t_F(\mu) + t_{\mathcal{E}}(\mu_e)$, asks at most q_e queries to its oracle, with total length at most μ_e.

Proof. Applying standard techniques, we expand $\mathbf{Adv}^{\mathrm{nAE}}_{\boldsymbol{\mathcal{E}}}(\mathcal{A})$ as follows.

$$\begin{aligned}
\mathbf{Adv}^{\mathrm{nAE}}_{\boldsymbol{\mathcal{E}}}(\mathcal{A}) &\le \Pr\left[\mathcal{A}^{\boldsymbol{\mathcal{E}}[\mathcal{E},F^{\mathrm{iv}},F^{\mathrm{tag}}],\boldsymbol{\mathcal{D}}[\mathcal{E},F^{\mathrm{iv}},F^{\mathrm{tag}}]} \Rightarrow 1\right] - \Pr\left[\mathcal{A}^{\boldsymbol{\mathcal{E}}[\mathcal{E},\rho_{\mathrm{iv}},\rho_{\mathrm{tag}}],\boldsymbol{\mathcal{D}}[\mathcal{E},\rho_{\mathrm{iv}},\rho_{\mathrm{tag}}]} \Rightarrow 1\right] \qquad (4)\\
&\quad + \Pr\left[\mathcal{A}^{\boldsymbol{\mathcal{E}}[\mathcal{E},\rho_{\mathrm{iv}},\rho_{\mathrm{tag}}],\boldsymbol{\mathcal{D}}[\mathcal{E},\rho_{\mathrm{iv}},\rho_{\mathrm{tag}}]} \Rightarrow 1\right] - \Pr\left[\mathcal{A}^{\$\boldsymbol{\mathcal{E}},\perp} \Rightarrow 1\right]\\
&\le \mathbf{Adv}^{\mathrm{prf}}_{F}(\mathcal{B}(\mathcal{A})) + \mathbf{Adv}^{\mathrm{nAE}}_{\tilde{\boldsymbol{\mathcal{E}}}}(\mathcal{A})
\end{aligned}$$

where the probabilities are over the implicit sampling of keys K, L or random functions ρ_{iv}, ρ_{tag}. When the compositional method is A4, $\rho_{\text{iv}} = \rho_{\text{tag}}$. For compactness, $\$\boldsymbol{\mathcal{E}}$ is the random bits oracle for the nAE-advantage over $\boldsymbol{\mathcal{E}}$. (This is to distinguish it from the random-bits oracle \$ for the ivE-advantage over $\mathcal{E}$.) The difference of probabilities in line (4) is bounded by $\mathbf{Adv}_F^{\text{prf}}(\mathcal{B}(\mathcal{A}))$, where $\mathcal{B}$ runs $\mathcal{A}$ and responds to its oracle queries as follows. When $\mathcal{A}$ asks encryption query (N, A, M), $\mathcal{B}$ calls its oracle once with the inputs needed to create IV, and a second time (if necessary) with the inputs needed to create a tag T. It then computes $C \leftarrow \mathcal{E}_K(IV, M \,\|\, T)$ or $C \leftarrow \mathcal{E}_K(IV, M) \,\|\, T$ as appropriate for the composition, and returns C to $\mathcal{A}$. When $\mathcal{A}$ asks decryption query (N, A, C), $\mathcal{B}$ similarly simulates $\boldsymbol{\mathcal{D}}$, using its oracle to produce the needed IV and to check the validity of the tag.

We now bound $\mathbf{Adv}_{\tilde{\boldsymbol{\mathcal{E}}}}^{\text{nAE}}(\mathcal{A})$. To this end, we continue our expansion

$$\begin{aligned}\mathbf{Adv}_{\tilde{\boldsymbol{\mathcal{E}}}}^{\text{nAE}}(\mathcal{A}) &= \Pr\left[\mathcal{A}^{\boldsymbol{\mathcal{E}}[\mathcal{E},\rho_{\text{iv}},\rho_{\text{tag}}],\boldsymbol{\mathcal{D}}[\mathcal{E},\rho_{\text{iv}},\rho_{\text{tag}}]} \Rightarrow 1\right] - \Pr\left[\mathcal{A}^{\boldsymbol{\mathcal{E}}[\mathcal{E},\rho_{\text{iv}},\rho_{\text{tag}}],\perp} \Rightarrow 1\right] \\ &\quad + \Pr\left[\mathcal{A}^{\boldsymbol{\mathcal{E}}[\mathcal{E},\rho_{\text{iv}},\rho_{\text{tag}}],\perp} \Rightarrow 1\right] - \Pr\left[\mathcal{A}^{\boldsymbol{\mathcal{E}}[\$,\rho_{\text{iv}},\rho_{\text{tag}}],\perp} \Rightarrow 1\right] \\ &\quad + \Pr\left[\mathcal{A}^{\boldsymbol{\mathcal{E}}[\$,\rho_{\text{iv}},\rho_{\text{tag}}],\perp} \Rightarrow 1\right] - \Pr\left[\mathcal{A}^{\$\boldsymbol{\mathcal{E}},\perp} \Rightarrow 1\right] \\ &\leq \Pr\left[\mathcal{A}^{\boldsymbol{\mathcal{E}}[\mathcal{E},\rho_{\text{iv}},\rho_{\text{tag}}],\boldsymbol{\mathcal{D}}[\mathcal{E},\rho_{\text{iv}},\rho_{\text{tag}}]} \Rightarrow 1\right] - \Pr\left[\mathcal{A}^{\boldsymbol{\mathcal{E}}[\mathcal{E},\rho_{\text{iv}},\rho_{\text{tag}}],\perp} \Rightarrow 1\right] + \mathbf{Adv}_{\mathcal{E}}^{\text{ivE}}(\mathcal{D}_1(\mathcal{A}))\end{aligned}$$

where $\Pr\left[\mathcal{A}^{\boldsymbol{\mathcal{E}}[\$,\rho_{\text{iv}},\rho_{\text{tag}}],\perp} \Rightarrow 1\right] - \Pr\left[\mathcal{A}^{\$\boldsymbol{\mathcal{E}},\perp} \Rightarrow 1\right] = 0$ for any $\mathcal{A}$. For compositional methods A1-A3 and A5-A9, the reduction $\mathcal{D}_1$ runs $\mathcal{A}$ and response to oracle queries as follows. When $\mathcal{A}$ makes encryption query (N, A, M), $\mathcal{D}_1$ creates tag T by simulating ρ_{tag} locally (via lazy sampling), using the appropriate input for the composition. It then calls its own oracle, on M or $M \,\|\, T$ (as appropriate), to simulate the remainder of $\boldsymbol{\mathcal{E}}$. We note that since the nonce N is always part of the input to ρ_{iv}, the uniformly random IVs returned by either the $\mathcal{E}_K$- or \$-oracles suffices to simulate ρ_{iv}. When A asks a decryption query, $\mathcal{D}_1$ simply returns $\perp$.

For compositional method A4, the reduction $\mathcal{D}_1$ behaves similarly, except that it uses the oracle-returned IVs as both IV and T.

Finally, we claim that $\Pr\left[\mathcal{A}^{\boldsymbol{\mathcal{E}}[\mathcal{E},\rho_{\text{iv}},\rho_{\text{tag}}],\boldsymbol{\mathcal{D}}[\mathcal{E},\rho_{\text{iv}},\rho_{\text{tag}}]} \Rightarrow 1\right] - \Pr\left[\mathcal{A}^{\boldsymbol{\mathcal{E}}[\mathcal{E},\rho_{\text{iv}},\rho_{\text{tag}}],\perp} \Rightarrow 1\right] \leq \mathbf{Adv}_{\tilde{\boldsymbol{\mathcal{E}}}}^{\text{auth}}(\mathcal{A})$. To see this, note that the encryption oracles in the probabilities on the left side of claimed expression is identical to the one provided by the experiment defining $\mathbf{Adv}_{\tilde{\boldsymbol{\mathcal{E}}}}^{\text{auth}}(\cdot)$. Moreover, the difference on the left side is 0 unless $\mathcal{A}$ makes a decryption query (N, A, C) that would cause $\boldsymbol{\mathcal{D}}[\mathcal{E}, \rho_{\text{iv}}, \rho_{\text{tag}}]$ to return something other than the symbol $\perp$. But this is exactly the event that would cause the decryption oracle $\boldsymbol{\mathcal{D}}^*[\mathcal{E}, \rho_{\text{iv}}, \rho_{\text{tag}}]$ to return 1 in the Auth-experiment. Since we can assume without loss that $\mathcal{A}$ halts with its output immediately upon receiving something other than $\perp$ from its decryption oracle, the claim is proved. The proof of the lemma follows. □

A.2 Proof of Security for A1, A2, A3, A5, A6, A7, A8

In light of Lemma 3, we concern ourselves with bounding $\mathbf{Adv}_{\tilde{\boldsymbol{\mathcal{E}}}}^{\text{auth}}(\mathcal{A})$. We can assume without loss that $\mathcal{A}$ asks a single decryption query, this being its final query before terminating its execution. By Lemma 2 we have $\mathbf{Adv}_{\tilde{\boldsymbol{\mathcal{E}}}}^{\text{auth}}(\mathcal{A}) \leq q_d \cdot \mathbf{Adv}_{\tilde{\boldsymbol{\mathcal{E}}}}^{\text{auth1}}(\mathcal{A})$, with a mild abuse of notation. We will refer to the single decryption query made by $\mathcal{A}$ as the *forgery attempt*.

Let the encryption queries of $\mathcal{A}$ be (N_i, A_i, M_i), and C_i the corresponding ciphertexts, for each $i \in [q_e]$, and let C_i be the corresponding ciphertexts. For each of these queries, let $V_i \in \mathcal{X}^{\text{iv}}$ be the input to F^{iv}. Let (N^*, A^*, C^*) be the forgery attempt, and let $V^* \in \mathcal{X}^{\text{iv}}$ be defined as were the V_i, so we can define $IV^* = \rho_{\text{iv}}(V^*)$ as the IV associated to the forgery attempt. We break the remainder of our analysis into two pieces, first considering A1–A6, and then A7–A8.

Schemes A1–A3, A5–A6. For these parse $C^* = Y^* \,\|\, T^*$, where $|T^*| = \tau$, and define define M^* to be the unique string such that $Y^* = \mathcal{E}_K(IV^*, M^*)$. (Here, we have used the tidiness of $\mathcal{E}$ to equate decryption of Y^* with the encryption of M^*.) To bound $\mathbf{Adv}^{\text{auth1}}_{\tilde{\mathcal{E}}}(\mathcal{A})$ there are two cases to consider.

Case 1: For A1–A3, assume $(N^*, A^*, M^*) \neq (N_i, A_i, M_i)$ for any $i \in [q_e]$. Then the forgery attempt is valid with probability at most $1/2^\tau$ because $\rho_{\text{tag}}(N^*, A^*, M^*)$ is uniformly random.

Likewise for A5–A6, assume $(N^*, A^*, C^*) \neq (N_i, A_i, C_i)$ for any $i \in [q_e]$. Then the forgery attempt is valid with probability at most $1/2^\tau$ because $\rho_{\text{tag}}(N^*, A^*, C^*)$ is uniformly random.

Case 2: For A1–A3, assume that $(N^*, A^*, M^*) = (N_j, A_j, M_j)$ for some particular $j \in [q_e]$. (There can be only one such j, since N_j is distinct.) Note that this implies $T^* = T_j$, too. Moreover, we claim that $V^* \neq V_j$. To see this, assume that $V^* = V_j$, and notice that this implies $IV* = \rho_{\text{iv}}(V_j) = IV_j$. Thus the forgery attempt is valid iff $C^* = \mathcal{E}(IV_j, M_j) \,\|\, T_j = C_j$; but then $(N^*, A^*, C^*) = (N_j, A_j, C_j)$ is not an allowed forgery attempt. So it must be that $V^* \neq V_j$. But this is impossible, since the assumption of the case is that $(N^*, A^*, M^*) = (N_j, A_j, M_j)$, and this implies that $V^* = V_j$. So this case never results in a valid forgery attempt for A1–A4.

For A5–A6 there is nothing to consider in this case, since $(N^*, A^*, C^*) = (N_j, A_j, C_j)$ for some $j \in [q_e]$ would be a disallowed forgery attempt.

Thus, for A1–A3 and A5–A6 we can conclude that $\mathbf{Adv}^{\text{auth1}}_{\tilde{\mathcal{E}}}(\mathcal{A}) \leq 1/2^\tau$.

Schemes A7–A8. Define $M^* \,\|\, T^*$ to be the unique string such that $C^* = \mathcal{E}_K(IV^*, M^* \,\|\, T^*)$, where $|T^*| = \tau$. Again, to bound $\mathbf{Adv}^{\text{auth1}}_{\tilde{\mathcal{E}}}(\mathcal{A})$ there are two cases to consider.

Case 1: Assume $(N^*, A^*, M^*) \neq (N_i, A_i, M_i)$ for any $i \in [q_e]$. Then the forgery attempt is valid with probability at most $1/2^\tau$ because $\rho_{\text{tag}}(N^*, A^*, M^*)$ is uniformly random.

Case 2: Assume $(N^*, A^*, M^*) = (N_j, A_j, M_j)$ for some particular $i \in [q_e]$, with $T^* = T_j$ as a consequence. It must be that $V^* \neq V_j$, since otherwise $IV^* = IV_j$ and $C^* = \mathcal{E}_K(IV_j, M_j \,\|\, T_j) = C_j$, and this would result in $(N^*, A^*, C^*) = (N_j, A_j, C_j)$ being a disallowed forgery attempt. But the assumption of the case makes $V^* \neq V_j$ is impossible. So this case never results in a valid forgery attempt.

Thus, for A7–A8 we can also conclude that $\mathbf{Adv}^{\text{auth1}}_{\tilde{\mathcal{E}}}(\mathcal{A}) \leq 1/2^\tau$.

Pulling everything together and employing Lemma 3, we have $\mathbf{Adv}^{\text{nAE}}_{\tilde{\mathcal{E}}}(\mathcal{A}) \leq \mathbf{Adv}^{\text{prf}}_{F}(\mathcal{B}(\mathcal{A})) + \mathbf{Adv}^{\text{ivE}}_{\mathcal{E}}(\mathcal{D}(\mathcal{A})) + q_d/2^\tau$.

A.3 Proof of Security for A4

In light of Lemma 3, we concern ourselves with bounding $\mathbf{Adv}^{\text{auth}}_{\tilde{\mathcal{E}}}(\mathcal{A})$. We can assume without loss that $\mathcal{A}$ asks a single decryption query, this being its final query before terminating its execution. By Lemma 2 we have $\mathbf{Adv}^{\text{auth}}_{\tilde{\mathcal{E}}}(\mathcal{A}) \leq q_d \cdot \mathbf{Adv}^{\text{auth1}}_{\tilde{\mathcal{E}}}(\mathcal{A})$, with a mild abuse of notation. We will refer to the single decryption query made by $\mathcal{A}$ as the *forgery attempt*.

Let the encryption queries of $\mathcal{A}$ be (N_i, A_i, M_i), and C_i the corresponding ciphertexts, for each $i \in [q_e]$, and let C_i be the corresponding ciphertexts. Let (N^*, A^*, C^*) be the forgery attempt, and define $IV^* = \rho(N^*, A^*, M^*)$ as the IV associated to the forgery attempt. Parse $C^* = Y^* \,\|\, T^*$, where $|T^*| = n$, and define define M^* to be the unique string such that $Y^* = \mathcal{E}_K(IV^*, M^*)$. (Here, we have used the tidiness of $\mathcal{E}$ to equate decryption of Y^* with the encryption of M^*.) To bound $\mathbf{Adv}^{\text{auth1}}_{\tilde{\boldsymbol{\mathcal{E}}}}(\mathcal{A})$ there are two cases to consider.

Case 1: Assume $(N^*, A^*, M^*) \neq (N_i, A_i, M_i)$ for any $i \in [q_e]$. Then the forgery attempt is valid with probability at most $1/2^n$ because $\rho(N^*, A^*, M^*)$ is uniformly random.

Case 2: Assume that $(N^*, A^*, M^*) = (N_j, A_j, M_j)$ for some particular $j \in [q_e]$. (There can be only one such j, since N_j is distinct.) Note that this implies $T^* = T_j$, and $IV^* = IV_j$. But then it must be that $C^* = C_j$, in which case $(N^*, A^*, C^*) = (N_j, A_j, C_j)$ is not a valid forgery attempt. So this case never results in a valid forgery attempt.

Thus, for A4 we can conclude that $\mathbf{Adv}^{\text{auth1}}_{\tilde{\boldsymbol{\mathcal{E}}}}(\mathcal{A}) \leq 1/2^n$.
Pulling everything together and employing Lemma 3, we have $\mathbf{Adv}^{\text{nAE}}_{\tilde{\boldsymbol{\mathcal{E}}}}(\mathcal{A}) \leq \mathbf{Adv}^{\text{prf}}_{F}(\mathcal{B}(\mathcal{A})) + \mathbf{Adv}^{\text{ivE}}_{\mathcal{E}}(\mathcal{D}(\mathcal{A})) + q_d/2^n$.

A.4 Proof of Security for A9

In light of Lemma 3, we concern ourselves with bounding $\mathbf{Adv}^{\text{auth}}_{\tilde{\boldsymbol{\mathcal{E}}}}(\mathcal{A})$. We can assume without loss that $\mathcal{A}$ asks a single decryption query, this being its final query before terminating its execution. By Lemma 2 we have $\mathbf{Adv}^{\text{auth}}_{\tilde{\boldsymbol{\mathcal{E}}}}(\mathcal{A}) \leq q_d \cdot \mathbf{Adv}^{\text{auth1}}_{\tilde{\boldsymbol{\mathcal{E}}}}(\mathcal{A})$, with a mild abuse of notation. We will refer to the single decryption query made by $\mathcal{A}$ as the *forgery attempt*. Let the encryption queries of $\mathcal{A}$ be (N_i, A_i, M_i) with $i \in [q_e]$. Let (N^*, A^*, C^*) be the forgery attempt, denote by $IV^* = \rho_{\text{iv}}(N^*, A^*)$ be the associated IV, and let $M^* \,\|\, T^*$ be the unique string such that $C^* = \mathcal{E}_K(IV^*, M^* \,\|\, T^*)$, where $|T^*| = \tau$. (Here we have used the tidiness of $\mathcal{E}$ to equate decryption of C^* with encryption of $M^* \,\|\, T^*$). To bound $\mathbf{Adv}^{\text{auth1}}_{\tilde{\boldsymbol{\mathcal{E}}}}(\mathcal{A})$ there are two cases to consider.

Case 1: $(N^*, M^*) \neq (N_i, M_i)$ for any $i \in [q_e]$. Then the forgery attempt is valid with probability at most $1/2^\tau$ because $\rho_{\text{tag}}(N^*, M^*)$ is uniformly random.

Case 2: $(N^*, M^*) = (N_j, M_j)$ for some $j \in [q_e]$. In this case, it must be that $T^* = T_j$, and we observe that N^* tells us exactly what is the index j. We claim that it must be that $A^* \neq A_j$. For if $A^* = A_j$, then $IV^* = \rho_{\text{iv}}(N_j, A_j) = IV_j$, and the forgery attempt is valid iff $C^* = \mathcal{E}(IV_j, M_j \,\|\, T_j) = C_j$. But then $(N^*, A^*, C^*) = (N_j, A_j, C_j)$ which is disallowed.

Now, $A^* \neq A_j$ implies that $IV^* = \rho_{\text{iv}}(N_j, A^*)$ is uniformly random. Moreover, the forgery attempt is valid iff $C^* = \mathcal{E}(IV^*, M_j \,\|\, T_j)$ for a random IV. Let $\mathcal{D}_2^{\text{Enc}}(\mathcal{A})$ be an ivE-adversary (for $\mathcal{E}$) that operates as follows. When $\mathcal{A}$ makes query (N_i, A_i, M_i), adversary $\mathcal{D}_2(\mathcal{A})$ computes $T_i = \rho_{\text{tag}}(N_j, M_j)$ via lazy sampling, and queries $\text{Enc}(M_i \,\|\, T_i)$. This returns $IV_i \,\|\, C_i$, and $\mathcal{D}_2(\mathcal{A})$ returns C_i to $\mathcal{A}$. Adversary $\mathcal{D}_2(\mathcal{A})$ stores $(N_i, A_i, M_i, T_i, C_i)$ for later use. When $\mathcal{A}$ halts with its forgery attempt (N^*, A^*, C^*), adversary $\mathcal{D}_2(\mathcal{A})$ finds the $N_j = N^*$ and then (re)queries $\text{Enc}(M_j \,\|\, T_j)$. This returns $IV^* \,\|\, C$ and $\mathcal{D}_2(\mathcal{A})$ outputs 1 iff $C^* = C$.

When $\text{Enc}(X) = IV \,\|\, \mathcal{E}_K^{IV}(X)$, we see that the simulation of $\tilde{\boldsymbol{\mathcal{E}}} = \boldsymbol{\mathcal{E}}[\mathcal{E}, \rho_{\text{iv}}, \rho_{\text{tag}}]$ is correct because the N_i are all distinct (by assumption), meaning that the random IVs sampled by Enc faithfully implement random function ρ_{iv}. Thus $\Pr\left[\, \mathcal{D}_2^{\mathcal{E}}(\mathcal{A}) \Rightarrow 1 \,\right] \geq \Pr\left[\mathcal{A}^{\tilde{\boldsymbol{\mathcal{E}}}} \text{ forges} \right]$.

On the other hand, when $\mathrm{Enc}(X) = IV \,\|\, \$(X)$, we have $\Pr\left[\mathcal{D}_2^{\$}(\mathcal{A}) \Rightarrow 1\right] \le 1/2^\tau$. To see this, note that the ciphertext C is a random bit string of length at least τ bits (as are all of the C_i observed by $\mathcal{A}$). Thus, $\mathbf{Adv}^{\mathrm{ivE}}_{\mathcal{E}}(\mathcal{D}_2(\mathcal{A})) \ge \Pr\left[\mathcal{A}^{\tilde{\boldsymbol{\mathcal{E}}}} \text{ forges}\right] - 1/2^\tau$ in this case.

As Case 1 and 2 are mutually exclusive and exhaustive, we have $\mathbf{Adv}^{\mathrm{auth1}}_{\tilde{\boldsymbol{\mathcal{E}}}}(\mathcal{A}) \le \mathbf{Adv}^{\mathrm{ivE}}_{\mathcal{E}}(\mathcal{D}_2(\mathcal{A})) + 2/2^\tau$. Pulling everything together and employing Lemma 3, we have $\mathbf{Adv}^{\mathrm{nAE}}_{\boldsymbol{\mathcal{E}}}(\mathcal{A}) \le \mathbf{Adv}^{\mathrm{prf}}_{F}(\mathcal{B}(\mathcal{A})) + \mathbf{Adv}^{\mathrm{ivE}}_{\mathcal{E}}(\mathcal{D}_1(\mathcal{A})) + q_d\left(\mathbf{Adv}^{\mathrm{ivE}}_{\mathcal{E}}(\mathcal{D}_2(\mathcal{A})) + 2/2^\tau\right)$.

B Proofs of Security for N-Schemes

B.1 Proof Preliminiaries for the proofs for N-Schemes

The proofs for N1–N3 all share a common opening in a similar manner as A1–A10. We capture this commonality in the following lemma.

Lemma 4. Fix a compositional method $\mathrm{Nn} \in \{\mathrm{N1}, \mathrm{N2}, \mathrm{N3}\}$, and let $\mathcal{E}\colon \mathcal{K} \times \{0,1\}^\eta \times \mathcal{M} \to \{0,1\}^*$ be an nE-scheme. Fix integers $0 < \eta, \tau \le r$, and let $F\colon \mathcal{L} \times \mathcal{X} \to \{0,1\}^r$ be a vecMAC. From F, derive $F^{\mathrm{tag}}\colon \mathcal{L} \times \mathcal{X}^{\mathrm{tag}} \to \{0,1\}^\tau$ according to our conventions, so that the resulting nAE-scheme $\boldsymbol{\mathcal{E}} = \mathrm{Nn}[\mathcal{E}, F^{\mathrm{tag}}]$ is syntactically valid. Let $\mathcal{A}$ be an nAE-adversary for $\boldsymbol{\mathcal{E}}$. Then there are blackbox reductions $\mathcal{B}$ and $\mathcal{D}_1$, explicitly given and analyzed in the proof of this theorem, such that

$$\mathbf{Adv}^{\mathrm{nAE}}_{\boldsymbol{\mathcal{E}}}(\mathcal{A}) \le \mathbf{Adv}^{\mathrm{prf}}_{F}(\mathcal{B}(\mathcal{A})) + \mathbf{Adv}^{\mathrm{auth}}_{\tilde{\boldsymbol{\mathcal{E}}}}(\mathcal{A}) + \mathbf{Adv}^{\mathrm{nE}}_{\mathcal{E}}(\mathcal{D}_1(\mathcal{A}))$$

where $\tilde{\boldsymbol{\mathcal{E}}}$ is shorthand for the composition $\mathrm{Nn}[\mathcal{E}, \rho_{\mathrm{tag}}]$ using random function ρ_{tag} in place of F^{tag}. Say $\mathcal{A}$ asks q_e queries to its encryption oracle, and q_d queries to its decryption oracle, with $q = q_e + q_d$. The encryption queries have total length μ_e, and the decryption queries have total length μ_d, with $\mu = \mu_e + \mu_d$. Let t_A be the running time of A, given this total number and length of queries. Then reduction $\mathcal{B}$ has running time at most $t_A + 2t_F(\mu) + t_{\mathcal{E}}(\mu)$, asks at most $2(q_e + q_d)$ queries to its oracle, with total length at most μ. Reduction $\mathcal{D}_1$ has running time $t_A + t_F(\mu) + t_{\mathcal{E}}(\mu_e)$, asks at most q_e queries to its oracle, with total length at most μ_e.

Proof. Applying standard techniques, we expand $\mathbf{Adv}^{\mathrm{nAE}}_{\boldsymbol{\mathcal{E}}}(\mathcal{A})$ as follows.

$$\begin{aligned}\mathbf{Adv}^{\mathrm{nAE}}_{\boldsymbol{\mathcal{E}}}(\mathcal{A}) &\le \Pr\left[\mathcal{A}^{\boldsymbol{\mathcal{E}}[\mathcal{E},F^{\mathrm{tag}}],\boldsymbol{\mathcal{D}}[\mathcal{E},F^{\mathrm{tag}}]} \Rightarrow 1\right] - \Pr\left[\mathcal{A}^{\boldsymbol{\mathcal{E}}[\mathcal{E},\rho_{\mathrm{tag}}],\boldsymbol{\mathcal{D}}[\mathcal{E},\rho_{\mathrm{tag}}]} \Rightarrow 1\right] &(5)\\ &\quad+\Pr\left[\mathcal{A}^{\boldsymbol{\mathcal{E}}[\mathcal{E},\rho_{\mathrm{tag}}],\boldsymbol{\mathcal{D}}[\mathcal{E},\rho_{\mathrm{tag}}]} \Rightarrow 1\right] - \Pr\left[\mathcal{A}^{\$\boldsymbol{\mathcal{E}},\perp} \Rightarrow 1\right]\\ &\le \mathbf{Adv}^{\mathrm{prf}}_{F}(\mathcal{B}(\mathcal{A})) + \mathbf{Adv}^{\mathrm{nAE}}_{\tilde{\boldsymbol{\mathcal{E}}}}(\mathcal{A})\end{aligned}$$

where the probabilities are over the implicit sampling of keys K, L or random function ρ_{tag}. For compactness, $\$\boldsymbol{\mathcal{E}}$ is the random bits oracle for the nAE-advantage over $\boldsymbol{\mathcal{E}}$. (This is to distinguish it from the random-bits oracle \$ for the nE-advantage over $\mathcal{E}$.) The difference of probabilities in line (5) is bounded by $\mathbf{Adv}^{\mathrm{prf}}_{F}(\mathcal{B}(\mathcal{A}))$, where $\mathcal{B}$ runs $\mathcal{A}$ and responds to its oracle queries as follows. When $\mathcal{A}$ asks encryption query (N, A, M), $\mathcal{B}$ calls its oracle with the inputs needed to create a tag T. It then computes $C \leftarrow \mathcal{E}_K(N, M \,\|\, T)$ or $C \leftarrow \mathcal{E}_K(N, M) \,\|\, T$ as appropriate for the composition, and returns C to $\mathcal{A}$. When $\mathcal{A}$ asks decryption query (N, A, C), $\mathcal{B}$ similarly simulates $\boldsymbol{\mathcal{D}}$, using its oracle to check the validity of the tag.

We now bound $\mathbf{Adv}^{\mathrm{nAE}}_{\tilde{\boldsymbol{\mathcal{E}}}}(\mathcal{A})$. To this end, we continue our expansion

$$\begin{aligned}\mathbf{Adv}^{\mathrm{nAE}}_{\tilde{\boldsymbol{\mathcal{E}}}}(\mathcal{A}) &= \Pr\left[\mathcal{A}^{\boldsymbol{\mathcal{E}}[\mathcal{E},\rho_{\mathrm{tag}}],\boldsymbol{\mathcal{D}}[\mathcal{E},\rho_{\mathrm{tag}}]} \Rightarrow 1\right] - \Pr\left[\mathcal{A}^{\boldsymbol{\mathcal{E}}[\mathcal{E},\rho_{\mathrm{tag}}],\perp} \Rightarrow 1\right] \\ &\quad + \Pr\left[\mathcal{A}^{\boldsymbol{\mathcal{E}}[\mathcal{E},\rho_{\mathrm{tag}}],\perp} \Rightarrow 1\right] - \Pr\left[\mathcal{A}^{\boldsymbol{\mathcal{E}}[\$,\rho_{\mathrm{tag}}],\perp} \Rightarrow 1\right] \\ &\quad + \Pr\left[\mathcal{A}^{\boldsymbol{\mathcal{E}}[\$,\rho_{\mathrm{tag}}],\perp} \Rightarrow 1\right] - \Pr\left[\mathcal{A}^{\$\boldsymbol{\mathcal{E}},\perp} \Rightarrow 1\right] \\ &\le \Pr\left[\mathcal{A}^{\boldsymbol{\mathcal{E}}[\mathcal{E},\rho_{\mathrm{tag}}],\boldsymbol{\mathcal{D}}[\mathcal{E},\rho_{\mathrm{tag}}]} \Rightarrow 1\right] - \Pr\left[\mathcal{A}^{\boldsymbol{\mathcal{E}}[\mathcal{E},\rho_{\mathrm{tag}}],\perp} \Rightarrow 1\right] + \mathbf{Adv}^{\mathrm{ivE}}_{\mathcal{E}}(\mathcal{D}_1(\mathcal{A}))\end{aligned}$$

where $\Pr\left[\mathcal{A}^{\boldsymbol{\mathcal{E}}[\$,\rho_{\mathrm{tag}}],\perp} \Rightarrow 1\right] - \Pr\left[\mathcal{A}^{\$\boldsymbol{\mathcal{E}},\perp} \Rightarrow 1\right] = 0$ for any $\mathcal{A}$. The reduction $\mathcal{D}_1$ runs $\mathcal{A}$ and responds to oracle queries as follows. When $\mathcal{A}$ makes encryption query (N, A, M), $\mathcal{D}_1$ creates tag T by simulating ρ_{tag} locally (via lazy sampling), using the appropriate input for the composition. It then calls its own oracle, on M or $M \,\|\, T$ (as appropriate), to simulate the remainder of $\boldsymbol{\mathcal{E}}$. When A asks a decryption query, $\mathcal{D}_1$ simply returns $\perp$.

Finally, we claim that $\Pr\left[\mathcal{A}^{\boldsymbol{\mathcal{E}}[\mathcal{E},\rho_{\mathrm{tag}}],\boldsymbol{\mathcal{D}}[\mathcal{E},\rho_{\mathrm{tag}}]} \Rightarrow 1\right] - \Pr\left[\mathcal{A}^{\boldsymbol{\mathcal{E}}[\mathcal{E},\rho_{\mathrm{tag}}],\perp} \Rightarrow 1\right] \le \mathbf{Adv}^{\mathrm{auth}}_{\tilde{\boldsymbol{\mathcal{E}}}}(\mathcal{A})$. To see this, note that the encryption oracles in the probabilities on the left side of claimed expression is identical to the one provided by the experiment defining $\mathbf{Adv}^{\mathrm{auth}}_{\tilde{\boldsymbol{\mathcal{E}}}}(\cdot)$. Moreover, the difference on the left side is 0 unless $\mathcal{A}$ makes a decryption query (N, A, C) that would cause $\boldsymbol{\mathcal{D}}[\mathcal{E}, \rho_{\mathrm{tag}}]$ to return something other than $\perp$. But this is exactly the event that would cause the decryption oracle $\boldsymbol{\mathcal{D}}^*[\mathcal{E}, \rho_{\mathrm{tag}}]$ to return 1 in the Auth-experiment. Since we can assume without loss that $\mathcal{A}$ halts with its output immediately upon receiving something other than $\perp$ from its decryption oracle, the claim is proved. The proof of the lemma follows. □

B.2 Proof of Security for N1, N2, N3

In light of Lemma 4, we concern ourselves with bounding $\mathbf{Adv}^{\mathrm{auth}}_{\tilde{\boldsymbol{\mathcal{E}}}}(\mathcal{A})$. We can assume without loss that $\mathcal{A}$ asks a single decryption query, this being its final query before terminating its execution. By Lemma 2 we have $\mathbf{Adv}^{\mathrm{auth}}_{\tilde{\boldsymbol{\mathcal{E}}}}(\mathcal{A}) \le q_d \cdot \mathbf{Adv}^{\mathrm{auth1}}_{\tilde{\boldsymbol{\mathcal{E}}}}(\mathcal{A})$, with a mild abuse of notation. We will refer to the single decryption query made by $\mathcal{A}$ as the *forgery attempt*.

Let the encryption queries of $\mathcal{A}$ be (N_i, A_i, M_i), and C_i the corresponding ciphertexts, for each $i \in [q_e]$. Let (N^*, A^*, C^*) be the forgery attempt. We break the remainder of our analysis into two pieces, first considering N1 and N2, then N3.

Schemes N1 and N2. For these parse $C^* = Y^* \,\|\, T^*$, where $|T^*| = n$, and define M^* to be the unique string such that $Y^* = \mathcal{E}_K(N^*, M^*)$. (Here, we have used the tidiness of $\mathcal{E}$ to equate decryption of Y^* with the encryption of M^*.) To bound $\mathbf{Adv}^{\mathrm{auth1}}_{\tilde{\boldsymbol{\mathcal{E}}}}(\mathcal{A})$ there are two cases to consider.

Case 1: For N1, assume $(N^*, A^*, M^*) \ne (N_i, A_i, M_i)$ for any $i \in [q_e]$. Then the forgery attempt is valid with probability at most $1/2^\tau$ because $\rho_{\mathrm{tag}}(N^*, A^*, M^*)$ is uniformly random.

Likewise for N2, assume $(N^*, A^*, C^*) \ne (N_i, A_i, C_i)$ for any $i \in [q_e]$. Then the forgery attempt is valid with probability at most $1/2^\tau$ because $\rho_{\mathrm{tag}}(N^*, A^*, C^*)$ is uniformly random.

Case 2: For N1, assume that $(N^*, A^*, M^*) = (N_j, A_j, M_j)$ for some particular $j \in [q_e]$. (There can be only one such j, since N_j is distinct.) Note that this implies $T^* = T_j$, too. Now notice that the forgery attempt is valid iff $C^* = \mathcal{E}(N_j, M_j) \,\|\, T_j = C_j$. Thus, $(N^*, A^*, C^*) = (N_j, A_j, C_j)$ and consequently is a disallowed forgery attempt.

Group	$\widetilde{\mathrm{A1}}$	$\widetilde{\mathrm{A2}}$	$\widetilde{\mathrm{A3}}$	$\widetilde{\mathrm{A4}}$	$\widetilde{\mathrm{A5}}$	$\widetilde{\mathrm{A6}}$	$\widetilde{\mathrm{A7}}$	$\widetilde{\mathrm{A8}}$	$\widetilde{\mathrm{A9}}$	$\widetilde{\mathrm{A10}}$
Schemes	$\mathrm{A}_1.\mathtt{0*****}$	$\mathrm{A}_1.\mathtt{1****0}$	$\mathrm{A}_1.\mathtt{1***01}$	$\mathrm{A}_1.\mathtt{1**011}$	$\mathrm{A}_2.\mathtt{0*0**1}$	$\mathrm{A}_2.\mathtt{1*0*01}$	$\mathrm{A}_2.\mathtt{1*0011}$	$\mathrm{A}_3.\mathtt{**0**0}$	$\mathrm{A}_3.\mathtt{0*0**1}$	$\mathrm{A}_3.\mathtt{100*01}$
Attack	Atk-1	Atk-2	Atk-3	Atk-4	Atk-1	Atk-5	Atk-6	Atk-7	Atk-8	Atk-9

Group	$\widetilde{\mathrm{N1}}$	$\widetilde{\mathrm{N2}}$	$\widetilde{\mathrm{N3}}$	$\widetilde{\mathrm{N4}}$	$\widetilde{\mathrm{N5}}$	$\widetilde{\mathrm{N7}}$	$\widetilde{\mathrm{N8}}$
Schemes	$\mathrm{N}_1.\mathtt{0**}$	$\mathrm{N}_1.\mathtt{10*}$	$\mathrm{N}_1.\mathtt{110}$	$\mathrm{N}_2.\mathtt{0*1}$	$\mathrm{N}_2.\mathtt{101}$	$\mathrm{N}_3.\mathtt{*0*}$	$\mathrm{N}_3.\mathtt{*10}$
Attack	Atk-10	Atk-11	Atk-2	Atk-6	Atk-12	Atk-13	Atk-14

Fig. 10. Insecure candidate schemes. The A-groups are constructed from an ivE and a vecMAC. The N-groups are constructed from an nE and a vecMAC. Recall that we only allow $\mathrm{A}_2.\mathtt{**0**1}$, $\mathrm{A}_3.\mathtt{**0***}$, and $\mathrm{N}_2.\mathtt{**1}$.

For N2, there is nothing to consider in this case, since $(N^*, A^*, C^*) = (N_j, A_j, C_j)$ for some $j \in [q_e]$ would be a disallowed forgery attempt.

Thus, we can conclude that $\mathbf{Adv}^{\mathrm{auth1}}_{\tilde{\boldsymbol{\mathcal{E}}}}(\mathcal{A}) \le 1/2^{\tau}$.

<u>Scheme N3.</u> Define $M^* \,\|\, T^*$ to be the unique string such that $C^* = \mathcal{E}_K(N^*, M^* \,\|\, T^*)$, where $|T^*| = n$. Again, to bound $\mathbf{Adv}^{\mathrm{auth1}}_{\tilde{\boldsymbol{\mathcal{E}}}}(\mathcal{A})$ there are two cases to consider.

Case 1: Assume $(N^*, A^*, M^*) \neq (N_i, A_i, M_i)$ for any $i \in [q_e]$. Then the forgery attempt is valid with probability at most $1/2^{\tau}$ because $\rho_{\mathrm{tag}}(N^*, A^*, M^*)$ is uniformly random.

Case 2: Assume $(N^*, A^*, M^*) = (N_j, A_j, M_j)$ for some particular $i \in [q_e]$, with $T^* = T_j$ as a consequence. Since $\boldsymbol{\mathcal{E}}$ is a deterministic function, it must be that $(N^*, A^*, C^*) = (N_j, A_j, C_j)$ and consequently is a disallowed forgery attempt.

Thus, for N3 we can also conclude that $\mathbf{Adv}^{\mathrm{auth1}}_{\tilde{\boldsymbol{\mathcal{E}}}}(\mathcal{A}) \le 1/2^{\tau}$.

Pulling everything together, we have $\mathbf{Adv}^{\mathrm{nAE}}_{\tilde{\boldsymbol{\mathcal{E}}}}(\mathcal{A}) \le \mathbf{Adv}^{\mathrm{prf}}_{F}(\mathcal{B}(\mathcal{A})) + \mathbf{Adv}^{\mathrm{nE}}_{\mathcal{E}}(\mathcal{D}_1(\mathcal{A})) + q_d/2^{\tau}$.

C Attacks on A-Schemes other than A1–A12

This section lists all the insecure schemes for the A-schemes along with the attacks against them as shown in Figure 10.

- **Atk-1:** Ask (N_1, A, M), receive $C_1 \,\|\, T1$. Then, ask (N_2, A, M), receive $C_2 \,\|\, T2$. If $C_1 = C_2$, then the encryption oracle is real. This breaks the privacy of the composed scheme.
- **Atk-2:** Let $\mathcal{E} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be the counter-mode encryption scheme based on a block cipher of size n. Let M be a string of n bits. Ask (N, A, M), receive $C\|T$ where $|C| = n$. Forge $(N, A, \overline{C}\|T)$. Since the tag does not depend on the plaintext, the tag verification will be successful.
- **Atk-3:** Let $\mathcal{E} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be a length-preserving scheme that admits one-bit plaintexts, and has n-bit IVs where $n \gg 1$. Ask $(N, A, 0)$, receive $C = C' \,\|\, T$, where $|C'| = 1$. Submit as forgery (N, A', C). Note that the forgery wins if $\mathcal{E}_K(IV, 0) = C'$, since in this case the tag T is correct. By pigeonhole, for any key K it must be that $\max_{b\in\{0,1\}}\{\Pr_{IV}\,(\mathcal{E}_K(IV, 0) = b)\} \ge 1/2$ and hence the probability that (N, A', C) is a valid forgery is at least $1/4$.
- **Atk-4:** This attack is almost the same as Atk-3 except that, for the forgery, we modify N instead of A. Specifically, let $\mathcal{E} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be a length-preserving scheme that admits one-bit plaintexts,

and has n-bit IVs where $n \gg 1$. Ask $(N, A, 0)$, receive $C = C' \,\|\, T$, where $|C'| = 1$. Submit as forgery (N', A, C). Note that the forgery wins if $\mathcal{E}_K(IV, 0) = C'$, since in this case the tag T is correct. By pigeonhole, for any key K it must be that $\max_{b\in\{0,1\}}\{\Pr_{IV}\left(\mathcal{E}_K(IV, 0) = b\right)\} \geq 1/2$ and hence the probability that (N', A, C) is a valid forgery is at least $1/4$.

- **Atk-5:** Ask (N, A, M), receive $C = C' \,\|\, T$. Submit as forgery (N, A', C) where $A \neq A'$. Since (N, C) has not changed, (N, A', C) is a valid forgery.
- **Atk-6:** Ask (N, A, M), receive $C = C' \,\|\, T$. Submit as forgery (N', A, C) where $N \neq N'$. Since (A, C) has not changed, (N', A, C) is a valid forgery.
- **Atk-7:** Let $\mathcal{E} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be the counter-mode encryption scheme based on a block cipher E whose block size is n, and let F^{tag} be a block cipher of size n as well. Let M be a string of length n. Ask (N, A, M), receive $C[1]C[2]$ where $|C[1]| = |C[2]| = n$. Submit as forgery $(N, A, \overline{C[1]}C[2])$. We argue that this forgery is valid. We first explain what happens if both N and A are used to compute the tag, i.e. the case where the scheme is A$_3$.`**0110`. The analysis for the case where only N (resp. A) is used to compute the tag, i.e. A$_3$.`**0100` (resp. A$_3$.`**0010`), can be easily adapted by removing A (resp. N) from the input list of F^{tag} in the following argument.

  Let $T = F_L^{\text{tag}}(N, A)$ where L is the key used for the tag-computing PRF. Notice that under counter-mode encryption, $C[2] = E_K(||IV + 1||) \oplus T$ where $||x||$ denotes an n-bit binary representation of the integer x and IV is the value of the IV computed from only either or both N and A. Let the decryption $\mathcal{D}_K(IV, \overline{C[1]}C[2]) = M' \| T'$ where $|T'| = n$. Notice that $T' = E_K(||IV + 1||) \oplus C[2] = T$. Thus, the tag verification succeeds.
- **Atk-8:** Let $\mathcal{E} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be the counter-mode encryption scheme based on a block cipher of size n. Let M be a string of n bits. Ask (N, A, M), receive $C1$. Then, ask (N', A, M), receive $C2$. If the first n bits of $C1$ equal those of $C2$, then the encryption oracle is real. This breaks the privacy of the composed scheme.
- **Atk-9:** Ask (N, A, M), receive C. Submit as forgery (N, A', C) where $A \neq A'$. Since the IV depends only on N, its value is unchanged. Consequently, C decrypts to the original M and its tag, which in turn is also unchanged since it is computed based on only N and M.

D Attacks on N-Schemes other than N1–N4

This section lists all the insecure schemes for the N-schemes along with the attacks against them as shown in Figure 10.

- **Atk-10:** Ask (N_1, A, M), receive $C_1 \| T_1$. Ask (N_2, A, M) where $N_2 \neq N_1$, receive $C_2 \| T_2$. If $T_1 = T_2$, then the encryption oracle is real.
- **Atk-11:** Ask (N, A, M), receive $C = C' \,\|\, T$. Submit as forgery (N, A', C) where $A \neq A'$. In order to see that (N, A', C) is valid, notice that the decryption $\mathcal{D}_K(N, C')$ must decrypt to the original plaintext M (due to the correctness of the base encryption scheme). Thus, even if both N and M are to be used to compute the tag, the resulting tag will still be T.
- **Atk-12:** Ask (N, A, M), receive $C = C' \,\|\, T$. Submit as forgery (N, A', C) where $A \neq A'$. Since N and C' have not changed, the tag T, which depends only on N and C', remains valid.
- **Atk-13:** Ask (N, A, M), receive C. Submit as forgery (N, A', C) where $A \neq A'$. In order to see that (N, A', C) is valid, notice that the decryption $\mathcal{D}_K(N, C)$ must decrypt to the original input $M \| T$ where T is the tag part (due to the correctness of the base encryption scheme). Thus, even if both N and M are to be used to compute the tag, the resulting tag will still be T.

```
$\mathrm{KoT}_{\mathcal{E},\mathsf{Ext},T_{sel},\tau}(\mathcal{A})$:
$i \leftarrow 0$; $\mathsf{win} \leftarrow 0$
$K \twoheadleftarrow \mathcal{K}$
Run $\mathcal{A}^{\mathrm{Enc,Reveal,Test}}$
Return $\mathsf{win}$
```

```
Oracle $\mathrm{Enc}(N, A, M)$:
$i \leftarrow i + 1$
$(N_i, A_i, M_i) \leftarrow (N, A, M)$
$IV_i \twoheadleftarrow \{0,1\}^n$
$S_i \leftarrow T_{sel}(N_i, A_i, M_i)$
if $T[S_i] = \perp$ then
    $T[S_i] \twoheadleftarrow \{0,1\}^\tau$
$T_i \leftarrow T[S_i]$
$X_i \leftarrow M_i \,\|\, T_i$
$C_i \leftarrow \mathcal{E}_K(IV_i, X_i)$
$\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(i, IV_i, M_i, C_i)\}$
Return $(IV_i, C_i)$
```

```
Oracle $\mathrm{Reveal}(j)$:
$\mathcal{T} \leftarrow \mathcal{T} \cup \{(j, T_j)\}$
Return $T_j$
```

```
Oracle $\mathrm{Test}(j^*, C^*)$:
$X \leftarrow \mathsf{Ext}(j^*, C^*, \mathcal{Q}, \mathcal{T})$
$\mathsf{valid} \leftarrow \mathsf{xgood} \leftarrow 0$
if $\exists X_i$ such that
(1) $C^* = \mathcal{E}_K(IV_{j^*}, X_i)$ and
(2) $(\cdot, T_i) \notin \mathcal{T}$ and
(3) $X_i = X_{j^*}$
then
    $\mathsf{valid} \leftarrow 1$
    if $X = X_i$ then $\mathsf{xgood} \leftarrow 1$
if $\mathsf{valid} \wedge \neg\mathsf{xgood}$ then
    $\mathsf{win} \leftarrow 1$
    Return 1
Return 0
```

Fig. 11. The Knowledge-of-Tags (KoT) Experiment. Function $T_{sel}(u, v, w)$ selects a fixed, ordered subset of its inputs. Deterministic algorithm Ext is the plaintext extractor. Parameter τ is the tag-length.

– **Atk-14:** Let $\mathcal{E} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be the counter-mode encryption scheme based on a block cipher E whose block size is n, and let F^{tag} be a block cipher of size n as well. Let M be a string of length n. Ask (N, A, M), receive $C[1]C[2]$ where $|C[1]| = |C[2]| = n$. Submit as forgery $(N, A, \overline{C[1]}C[2])$. We argue that this forgery is valid. We first explain what happens if both N and A are used to compute the tag, i.e. the case where the scheme is N3.110. The analysis for the case where only A is used to compute the tag, i.e. N3.010, can be easily adapted by removing N from the input list of F^{tag} in the following argument.

  Let $T = F^{\mathrm{tag}}_L(N, A)$ where L is the key used for the tag-computing PRF. Notice that under counter-mode encryption, $C[2] = E_K(||N + 1||) \oplus T$ where $||x||$ denotes an n-bit binary representation of the integer x. Let the decryption $\mathcal{D}_K(N, \overline{C[1]}C[2]) = M' \| T'$ where $|T'| = n$. Notice that $T' = E_K(||N + 1||) \oplus C[2] = T$. Thus, the tag verification succeeds.

E Elusive Schemes A10–A12

As we discussed in Section 3, efforts to prove that A11 and A12 are secure against ciphertext forgeries, assuming only a PRF-secure vecMAC and an ivE-secure $\mathcal{E}$, have failed. But there is intuitive support for thinking these compositions should be secure: all observed ciphertexts are indistinguishable from random bits and, as a consequence, at least τ-bits of the input to $\mathcal{E}$, the tag, should be unknown to the attacker. Hence, producing a forgery (N^*, A^*, C^*) that yields a valid $M^* \,\|\, T^*$ should be difficult — if M^* has never been seen by F^{tag}, then the correct T^* is random; if M^* has been seen by F^{tag}, then you need to know what value of T^* to "target" under decryption. (Tidiness of $\mathcal{E}$ rules out simple forgeries that work no matter what value the tags take.)

To formalize this intuition, we introduce the *knowledge-of-tags* (KoT) experiment in Figure 11. "Knowing" a tag is captured by introducing a plaintext extractor Ext, a deterministic algorithm that takes as input all of the inputs explicitly available to the forging adversary and outputs a string X or $\perp$. The forger wins it if manages to produce a forgery that uses an old $IV^* = IV_j$ and an old $M^* \,\|\, T^* = M_i \,\|\, T_i$ for which it does not (explicitly) know T_i, and yet the extractor fails to determine this $M_i \,\|\, T_i$. Loosely speaking if the forger wins the KoT game, it has done so without (extractable) knowledge of tag T_i.

For an IV-based encryption scheme $\mathcal{E}$, a plaintext extractor Ext, a tag-length τ, and a tag-input selection function T_{sel}, we write $\mathbf{Adv}^{\mathrm{kot}}_{\mathcal{E},\mathsf{Ext},T_{sel},\tau}(\mathcal{A}) = \Pr\left[\,\mathrm{KoT}_{\mathcal{E},\mathsf{Ext},T_{sel},\tau}(A) = 1\,\right]$. for the KoT-advantage of adversary $\mathcal{A}$.

As Theorem 4 shows, if $\mathcal{E}$ is KoT-secure (in addition to being ivE-secure) and $F^{\mathrm{tag}}, F^{\mathrm{iv}}$ are PRF-secure, then A11 is secure against ciphertext forgeries. We skip directly to analyzing A11$[\mathcal{E}, \rho_{\mathrm{iv}}, \rho_{\mathrm{tag}}]$, since the theorem lifts to A11$[\mathcal{E}, F^{\mathrm{iv}}, F^{\mathrm{tag}}]$ by standard methods. We also focus on adversaries making a single forgery attempt (ie, Auth1-adversaries), in light of Lemma 2. Theorem 5 gives a similar result for the A12 composition method.

Although we do not explore the KoT notion here, we note that it is straightforward to construct extractors for common blockcipher modes, such as CTR and CBC mode. Thus, we expect common blockcipher modes to be provably KoT-secure.

Theorem 4. Fix $n, \tau > 0$. Let $\mathcal{X}^{\mathrm{iv}}$ and $\mathcal{X}^{\mathrm{tag}}$ be the domains-of-application for F^{iv} and F^{tag} (resp.) in the A11 composition method. Let $\rho_{\mathrm{iv}} \colon \mathcal{X}^{\mathrm{iv}} \to \{0,1\}^n$ and $\rho_{\mathrm{tag}} \colon \mathcal{X}^{\mathrm{tag}} \to \{0,1\}^\tau$ be random functions. Let $\mathcal{E} \colon \mathcal{K} \times \{0,1\}^* \to \{0,1\}^*$ be an ivE-scheme with length function ℓ. Let $\boldsymbol{\mathcal{E}} = $ A11$[\mathcal{E}, \rho_{\mathrm{iv}}, \rho_{\mathrm{tag}}]$ be the A11-composition and, correspondingly, let tag-input selection function $T_{sel}(N, A, M) = M$. Let Ext be a plaintext extractor for $\mathcal{E}$. Let $\mathcal{A}$ be an Auth-adversary, asking q queries to its oracle, prior to outputting its single forgery attempt. Then there exist black-box reductions $\mathcal{D}_1$, $\mathcal{D}_2$, and $\mathcal{B}$, all explicitly constructed in the proof of this theorem, such that

$$\mathbf{Adv}^{\mathrm{auth1}}_{\boldsymbol{\mathcal{E}}}(\mathcal{A}) \leq \frac{3q+1}{2^\tau} + 2\mathbf{Adv}^{\mathrm{ivE}}_{\mathcal{E}}(\mathcal{D}_1(\mathcal{A})) + \mathbf{Adv}^{\mathrm{ivE}}_{\mathcal{E}}(\mathcal{D}_2(\mathcal{A})) + \mathbf{Adv}^{\mathrm{kot}}_{\mathcal{E},\mathsf{Ext},T_{sel},\tau}(\mathcal{B}(\mathcal{A}))$$

Proof. We begin by establishing some notation. Let $(N_1, A_1, M_1), (N_2, A_2, M_2), \ldots, (N_q, A_q, M_q)$ be the sequence of queries asked by $\mathcal{A}$, and let $C_1, C_2, \ldots, C_q$ be the corresponding oracle responses. Let (N^*, A^*, C^*) be the adversary's forgery attempt, $IV^* = \rho_{\mathrm{iv}}(N^*, A^*)$ be the IV induced by N^*, A^*, and let $M^* \,\|\, T^* = \mathcal{D}_K(IV^*, C^*)$ where $|T^*| = n$. We say that (N^*, A^*, C^*) is a valid forgery if $\rho_{\mathrm{tag}}(M^*) = T^*$, i.e. that decryption under the A11-composition will return a non-$\perp$ value.

<u>Case 1:</u> Assume that $M^* \notin \{M_1, M_2, \ldots, M_q\}$. The probability that (N^*, A^*, C^*) is a valid forgery is at most $1/2^\tau$, since (N^*, A^*, C^*) iff $\rho_{\mathrm{tag}}(M^*) = T^*$.

<u>Case 2:</u> Now assume that $M^* = M_i$ for some $i \in [q]$. Then the forgery is valid only if $T^* = T_i$, so assume that holds. Since $\mathcal{E}$ is tidy, this implies that $C^* = \mathcal{E}_K(IV^*, M_i \,\|\, T_i)$. There are two cases to consider, that IV^* is "new", meaning that $(N^*, A^*) \notin \{(N_1, A_1), (N_2, A_2), \ldots, (N_q, A_q)\}$, and the complementary case.

<u>Case 2a:</u> Assume that $(N^*, A^*) \notin \{(N_1, A_1), (N_2, A_2), \ldots, (N_q, A_q)\}$. Then IV^* is uniformly random, so the probability that the forgery is valid is equal to $p = \Pr\left[\,IV^* \twoheadleftarrow \{0,1\}^n \colon \mathcal{E}_K(IV^*, M_i \,\|\, T_i)\,\right]$. We now give an adversary $\mathcal{D}_1(\mathcal{A})$ such that $p \leq .5\mathbf{Adv}^{\mathrm{ivE}}_{\mathcal{E}}(\mathcal{D}_1(\mathcal{A}))$. Specifically, let $\mathcal{D}_1^{\mathcal{O}}(\mathcal{A})$ be defined as follows. It runs $\mathcal{A}$, and when asks query (N_i, A_i, M_i), adversary $\mathcal{D}_1(\mathcal{A})$ creates $T_i = \rho_{\mathrm{tag}}(M_i)$ by lazy sampling simulation of ρ_{tag}. It then queries $M_i \,\|\, T_i$ to its oracle $\mathcal{O}$, receiving (IV_i, C_i) in return, and responds to $\mathcal{A}$ with C_i. When $\mathcal{A}$ halts with forgery attempt (N^*, A^*, C^*), adversary $\mathcal{D}_1(\mathcal{A})$ again queries each of $M_1 \,\|\, T_1, M_2 \,\|\, T_2, \ldots, M_q \,\|\, T_q$ to $\mathcal{O}$, receiving $(V_1, Y_1), (V_2, Y_2), \ldots, (V_q, Y_q)$ in return. If $C^* \in \{Y_1, \ldots, Y_q\}$, then $\mathcal{D}_1(\mathcal{A})$ halts with output bit 1; if not, $\mathcal{D}_1(\mathcal{A})$ halts with output bit 0.

Consider $\mathbf{Adv}^{\mathrm{ivE}}_{\mathcal{E}}(\mathcal{D}_1(\mathcal{A})) = \Pr\left[\mathcal{D}_1^{\mathcal{E}_K(\cdot)}(\mathcal{A}) \Rightarrow 1\right] - \Pr\left[\mathcal{D}_1^{\$(\cdot)}(\mathcal{A}) \Rightarrow 1\right]$. It is not hard to verify that $\Pr\left[\mathcal{D}_1^{\mathcal{E}_K(\cdot)}(\mathcal{A}) \Rightarrow 1\right] \geq .5p$. If $\mathcal{O} = \$$ then $Y_1, \ldots, Y_q$ are uniformly random, independent strings. Thus the probability that $C^* \in \{Y_1, \ldots, Y_q\}$ is at most $q/2^s$ where $s = \min_{i\in[q]} |Y_i| \geq \tau$. We conclude that in Case 2a, the probability of a valid forgery is at most $2\mathbf{Adv}^{\mathrm{ivE}}_{\mathcal{E}}(\mathcal{D}_1(\mathcal{A})) + 2q/2^{\tau}$.

Case 2b: Assume that $(N^*, A^*) = (N_j, A_j)$ for some $j \in [q]$. This implies that $IV^* = IV_j$, and that the forgery is "Case 2b valid" iff $C^* = \mathcal{E}_K(IV_j, M_i \,\|\, T_i)$. We note immediately that it must be that $M_i \,\|\, T_i \neq M_j \,\|\, T_j$, since equality would result in $C^* = C_j$ and the forgery attempt being disallowed. Let $\Pr\left[(N^*, A^*, C^*) \twoheadleftarrow \mathcal{A} \text{ Case 2b valid}\right]$ denote the probability of forging in Case 2b.

To bound $\Pr\left[(N^*, A^*, C^*) \twoheadleftarrow \mathcal{A}\colon \text{ Case 2b valid}\right]$, we consider a new game, $H0$, given in Figure 13. In this game, adversary $\mathcal{A}$ is provided with an encryption oracle Enc that faithfully simulates encryption under $\boldsymbol{\mathcal{E}}$, but additionally providing IV_j. It also performs some bookkeeping that does not affect the oracle's behavior. The game provides $\mathcal{A}$ with a new Reveal-oracle, which allows the adversary to query an index j, and receive the tag T_j in return. Thus, this oracle reveals to the adversary what was the tag associated to the j-th encryption query. Game $H0$ returns 1 exactly when the flag $\mathsf{win} = 1$, and this occurs iff $\mathcal{A}$ outputs (j^*, C^*) such that $C^* = \mathcal{E}_K(IV_j, X_i)$, subject to two restrictions: first, $X_i \neq X_{j^*}$ (preventing trivial wins, in which $C^* = C_{j^*}$); and second X_i must be a string whose value is not explicitly available to $\mathcal{A}$ via having made a Reveal-query. We point out that once $\mathsf{win} \leftarrow 1$ then the game returns 1 no matter what Ext returns

We note that $\mathcal{A}$ in $H0$ is free to make no Reveal-queries (and ignore the IVs that Enc provides), in which case $H0$ returns 1 exactly when $\mathcal{A}$ would produce a valid Case 2b forgery. We can assume, then that

$$\Pr\left[(N^*, A^*, C^*) \twoheadleftarrow \mathcal{A}\colon \text{ Case 2b valid}\right] \leq \Pr\left[H0(\mathcal{A}) = 1\right]$$

with a slight abuse of the notation, to avoid declaring a new adversary. By total probability, we have $\Pr\left[H0(\mathcal{A}) = 1\right] = \Pr\left[H0(\mathcal{A}) = 1 \wedge \mathsf{xgood}\right] + \Pr\left[H0(\mathcal{A}) = 1 \wedge \neg\mathsf{xgood}\right]$.

We now show that $\Pr\left[H0(\mathcal{A}) = 1 \wedge \mathsf{xgood}\right]$ can be bounded in terms of the ivE-advantage of some adversary $\mathcal{D}_2(\mathcal{A})$ attacking $\mathcal{E}$. Let $D_2^{\mathcal{O}}$ run $\mathcal{A}$ and respond to its queries as follows. When $\mathcal{A}$ asks Enc-query (N_i, A_i, M_i), adversary $\mathcal{D}_2(\mathcal{A})$ simulates Enc up through creating X_i. It then queries X_i to its oracle, receiving (IV_i, C_i) in return, and then continues to simulate the maintenance of set $\mathcal{Q}$, responding to $\mathcal{A}$ with (IV_i, C_i). If $\mathcal{A}$ queries Reveal(j), then $\mathcal{D}_2(\mathcal{A})$ exactly simulates Reveal. When $\mathcal{A}$ outputs (j^*, C^*), adversary $\mathcal{D}_2(\mathcal{A})$ runs $\mathsf{Ext}(j^*, C^*, \mathcal{Q}, \mathcal{T})$, this returning a string X or $\perp$. If Ext returns string $X = X_i$ such that $T_i \notin \mathcal{T}$, then $\mathcal{D}_2(\mathcal{A})$ halts with output bit 1. Otherwise, $\mathcal{D}_2(\mathcal{A})$ halts with output bit 0.

We claim that $\Pr\left[H0(\mathcal{A}) = 1 \wedge \mathsf{xgood}\right] \leq \Pr\left[\mathcal{D}_2^{\mathcal{E}_K}(\mathcal{A}) \Rightarrow 1\right]$. This is true because $\mathcal{D}_2(\mathcal{A})$ outputs one if the extractor manages to return any X_i such that $T_i \notin \mathcal{T}$, but $H0(\mathcal{A}) = 1 \wedge \mathsf{xgood}$ only occurs if the extractor returns a particular X_i meeting that restriction. Thus, we have

$$\Pr\left[H0(\mathcal{A}) = 1\right] \leq \mathbf{Adv}^{\mathrm{ivE}}_{\mathcal{E}}(\mathcal{D}_2(\mathcal{A})) + \Pr\left[\mathcal{D}_2^{\$}(\mathcal{A}) \Rightarrow 1\right] + \Pr\left[H0(\mathcal{A}) = 1 \wedge \neg\mathsf{xgood}\right].$$

Now, to bound $\Pr\left[\mathcal{D}_2(\mathcal{A})^{\$} \Rightarrow 1\right]$, consider game $G0$ in Figure 12. We see that $G0(\mathcal{A})$ returns 1 exactly when $D2$ returns 1. Moreover, the Enc-oracle in $G0$ returns (IV_i, C_i), where $IV_i \twoheadleftarrow \{0,1\}^n$ and $C_i \twoheadleftarrow \{0,1\}^{\ell(|M_i|+\tau)}$, which is exactly what a \$-oracle would return. The additional code inside the Enc-oracle is for bookkeeping purposes only. Finally, the Reveal-oracle in $G0$ behaves exactly as in $H0$, as expected by $\mathcal{A}$. Thus, we see that $\Pr\left[\mathcal{D}_2^{\$}(\mathcal{A}) \Rightarrow 1\right] \leq \Pr\left[G0_{\mathsf{Ext},T_{sel},\tau}(\mathcal{A}) = 1\right]$. Now, notice that in $G0$ the responses of the Enc-oracle are independent of the tags T_i, and depend on the adversary's queries only through $|M_i|$. Thus, we can move the sampling of tags from Enc to Reveal, which is when the tag values actually must be fixed. Moreover, notice that only the particular T_j requested by $\mathcal{A}$ in its Reveal-queries are needed to execute $\mathcal{A}$ and Ext. Thus we can move the sampling of all other tags until after *both* the adversary $\mathcal{A}$ and the extractor Ext have finished their executions. We implement these changes in game $G1$, and conclude that $\Pr[\,G0_{\mathsf{Ext},T_{sel},\tau}(\mathcal{A}) = 1\,] = \Pr[\,G1_{\mathsf{Ext},T_{sel},\tau}(\mathcal{A}) = 1\,]$. But in $G1$ it is clear that the chance that the extractor returns X that causes $\mathsf{xgood} \leftarrow 1$ (thus $G1$ returns 1) is at most $(q - |\mathcal{T}|)/2^\tau$; hence $\Pr[\,G1_{\mathsf{Ext},T_{sel},\tau}(\mathcal{A}) = 1\,] \leq q/2^\tau$.

Finally, to bound $\Pr[\,H0(\mathcal{A}) = 1 \wedge \neg\mathsf{xgood}\,]$ let $\mathcal{B}(\mathcal{A})$ be a KoT-adversary that runs $\mathcal{A}$, responding to $\mathcal{A}$'s Enc and Reveal queries using its own oracles of the same name. When $\mathcal{A}$ outputs (j^*, C^*), adversary $\mathcal{B}$ forwards this to its Test-oracle. By construction, we see that $\mathsf{win} = 1$ in the KoT-game exactly when $\mathsf{valid} \wedge \neg\mathsf{xgood}$ holds in $H0$. Thus, $\Pr[\,H0(\mathcal{A}) = 1 \wedge \neg\mathsf{xgood}\,] \leq \mathbf{Adv}^{\mathrm{kot}}_{\mathcal{E},\mathsf{Ext},T_{sel},\tau}(\mathcal{B}(\mathcal{A}))$.

Putting it all together, we have

$$\Pr[\,(N^*, A^*, C^*) \text{ valid}\,] \leq \mathbf{Adv}^{\mathrm{ivE}}_{\mathcal{E}}(\mathcal{D}_2(\mathcal{A})) + q/2^\tau + \mathbf{Adv}^{\mathrm{kot}}_{\mathcal{E},\mathsf{Ext},T_{sel},\tau}(\mathcal{B}(\mathcal{A}))$$

in Case 2b. Combining this with the result for Case 2a gives the bound on Case 2; further combining this with Case 1 gives the bound of the theorem. $\square$

Theorem 5. Fix $n, \tau > 0$. Let $\mathcal{X}^{\mathrm{iv}}$ and $\mathcal{X}^{\mathrm{tag}}$ be the domains-of-application for F^{iv} and F^{tag} (resp.) in the A11 composition method. Let $\rho_{\mathrm{iv}} \colon \mathcal{X}^{\mathrm{iv}} \to \{0,1\}^n$ and $\rho_{\mathrm{tag}} \colon \mathcal{X}^{\mathrm{tag}} \to \{0,1\}^\tau$ be random functions. Let $\mathcal{E} \colon \mathcal{K} \times \{0,1\}^* \to \{0,1\}^*$ be an ivE-scheme with length function ℓ. Let $\boldsymbol{\mathcal{E}} = \mathrm{A11}[\mathcal{E}, \rho_{\mathrm{iv}}, \rho_{\mathrm{tag}}]$ be the A11-composition and, correspondingly, let tag-input selection function $T_{sel}(N, A, M) = (A, M)$. Let Ext be a plaintext extractor for $\mathcal{E}$. Let $\mathcal{A}$ be an Auth-adversary, asking q queries to its oracle, prior to outputting its single forgery attempt. Then there exist black-box reductions $\mathcal{D}_1$, $\mathcal{D}_2$, and $\mathcal{B}$, all explicitly constructed in the proof of this theorem, such that

$$\mathbf{Adv}^{\mathrm{auth1}}_{\boldsymbol{\mathcal{E}}}(\mathcal{A}) \leq \frac{3q+1}{2^\tau} + 2\mathbf{Adv}^{\mathrm{ivE}}_{\mathcal{E}}(\mathcal{D}_1(\mathcal{A})) + \mathbf{Adv}^{\mathrm{ivE}}_{\mathcal{E}}(\mathcal{D}_2(\mathcal{A})) + \mathbf{Adv}^{\mathrm{kot}}_{\mathcal{E},\mathsf{Ext},T_{sel},\tau}(\mathcal{B}(\mathcal{A}))$$

Proof. The proof is nearly identical to that of Theorem 4, with the following small alternations. Case 1 assumes that $(A^*, M^*) \notin \{(A_1, M_1), \ldots, (A_q, M_q)\}$, with the same conclusion of the case. Case 2 then assumes that $(A^*, M^*) = (A_i, M_i)$ for some $j \in [q]$, and proceeds to consider subcases 2a and 2b. The former supposes that IV^* is new (i.e. N^* is new), and the latter that $IV^* = IV_j$ for some $j \in [q]$ with $M_j \,\|\, T_j \neq M_i \,\|\, T_i$. The conclusions of these subcases are same as those of Theorem 4. $\square$

```
$G0(\mathcal{A})$ :
$i \leftarrow 0$; $\mathsf{xgood} \leftarrow 0$
$K \twoheadleftarrow \mathcal{K}$
$(j^*, C^*) \leftarrow \mathcal{A}^{\mathrm{Enc,Reveal}}$
$X \leftarrow \mathsf{Ext}(j^*, C^*, \mathcal{Q}, \mathcal{T})$
if $\exists i$ such that $X = X_i \wedge T_i \notin \mathcal{T}$
then $\mathsf{xgood} \leftarrow 1$
Return $\mathsf{xgood}$
```

```
Oracle Enc$(N, A, M)$:
$i \leftarrow i + 1$; $(N_i, A_i, M_i) \leftarrow (N, A, M)$
$S_i \leftarrow T_{sel}(N_i, A_i, M_i)$
if $T[S_i] = \perp$ then $T[S_i] \twoheadleftarrow \{0,1\}^\tau$
$T_i \leftarrow T[S_i]$
$X_i \leftarrow M_i \,\|\, T_i$
$IV_i \twoheadleftarrow \{0,1\}^n$
$C_i \twoheadleftarrow \{0,1\}^{\ell(|M_i|+\tau)}$
$\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(i, IV_i, M_i, C_i)\}$
Return $(IV_i, C_i)$
```

```
Oracle Reveal$(j)$:
$\mathcal{T} \leftarrow \mathcal{T} \cup \{(j, T_j)\}$
Return $T_j$
```

```
$G1(\mathcal{A})$ :
$i \leftarrow 0$; $\mathsf{xgood} \leftarrow 0$
$K \twoheadleftarrow \mathcal{K}$
$(j^*, C^*) \leftarrow \mathcal{A}^{\mathrm{Enc,Reveal}}$
$X \leftarrow \mathsf{Ext}(j^*, C^*, \mathcal{Q}, \mathcal{T})$
for $j = 1$ to $q$ do
  if $T[S_j] = \perp$ then $T[S_j] \twoheadleftarrow \{0,1\}^\tau$
  $T_j \leftarrow T[S_j]$
  $X_j \leftarrow M_j \,\|\, T_j$
if $\exists i$ such that $X = X_i \wedge T_i \notin \mathcal{T}$
then $\mathsf{xgood} \leftarrow 1$
Return $\mathsf{xgood}$
```

```
Oracle Enc$(N, A, M)$:
$i \leftarrow i + 1$; $(N_i, A_i, M_i) \leftarrow (N, A, M)$
$S_i \leftarrow T_{sel}(N_i, A_i, M_i)$
$IV_i \twoheadleftarrow \{0,1\}^n$
$C_i \twoheadleftarrow \{0,1\}^{\ell(|M_i|+\tau)}$
$\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(i, IV_i, M_i, C_i)\}$
Return $(IV_i, C_i)$
```

```
Oracle Reveal$(j)$:
$S_j \leftarrow T_{sel}(N_j, A_j, M_j)$
if $T[S_j] = \perp$ then
  $T[S_j] \twoheadleftarrow \{0,1\}^\tau$
$T_j \leftarrow T[S_j]$
$\mathcal{T} \leftarrow \mathcal{T} \cup \{(j, T_j)\}$
Return $T[S_j]$
```

Fig. 12. Games for the proof of Theorem 4.

```
$H0(\mathcal{A})$ :
$i \leftarrow 0$
$\mathsf{valid} \leftarrow \mathsf{xgood} \leftarrow 0$
$K \twoheadleftarrow \mathcal{K}$
$(j^*, C^*) \leftarrow \mathcal{A}^{\mathrm{Enc,Reveal}}$
$X \leftarrow \mathsf{Ext}(j^*, C^*, \mathcal{Q}, \mathcal{T})$
if $X_i = M_i \,\|\, T_i$ such that
(1) $C^* = \mathcal{E}_K(IV_{j^*}, X_i)$ and
(2) $X_i \neq X_{j^*}$ and
(3) $T_i \notin \mathcal{T}$
then
   $\mathsf{valid} \leftarrow 1$
   if $X = X_i$ then $\mathsf{xgood} \leftarrow 1$
Return $\mathsf{valid}$
```

```
Oracle Enc$(N, A, M)$:
$i \leftarrow i + 1$; $(N_i, A_i, M_i) \leftarrow (N, A, M)$
$S_i \leftarrow T_{sel}(N_i, A_i, M_i)$
if $T[S_i] = \perp$ then $T[S_i] \twoheadleftarrow \{0,1\}^\tau$
$T_i \leftarrow T[S_i]$
$X_i \leftarrow M_i \,\|\, T_i$
$IV_i \twoheadleftarrow \{0,1\}^n$
$C_i \twoheadleftarrow \mathcal{E}_K(IV_i, X_i)$
$\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(i, IV_i, M_i, C_i)\}$
Return $(IV_i, C_i)$
```

```
Oracle Reveal$(j)$:
$\mathcal{T} \leftarrow \mathcal{T} \cup \{T_j\}$
Return $T_j$
```

Fig. 13. Games for the proof of Theorem 4.