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

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.

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 [4, 5] (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

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))
pAE	K, M	K, C	privacy + auth: ind, plus adv can't forge ciphertexts
ivE	K, IV, M	K, IV, C	privacy: $(IV_i \,\|\, C_i) \approx$ rand-bits
nE	K, N, M	K, N, C	privacy: ind\$ = (ciphertexts $\approx$ rand-bits)
nAE	K, N, A, M	K, N, A, C	privacy + auth: ind\$ + adv can't forge ciphertexts

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.

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 Fig. 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 [3, 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 detect *forged* ciphertexts [4–6].

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, 3] 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.

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 $\text{pE} + \text{MAC} \rightarrow \text{pAE}$ 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 $\text{ivE} + \text{MAC} \rightarrow \text{nAE}$ 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 Fig. 2. Two A-schemes are *transitional*: they have inferior established bounds. Two 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 Fig. 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 [7] (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

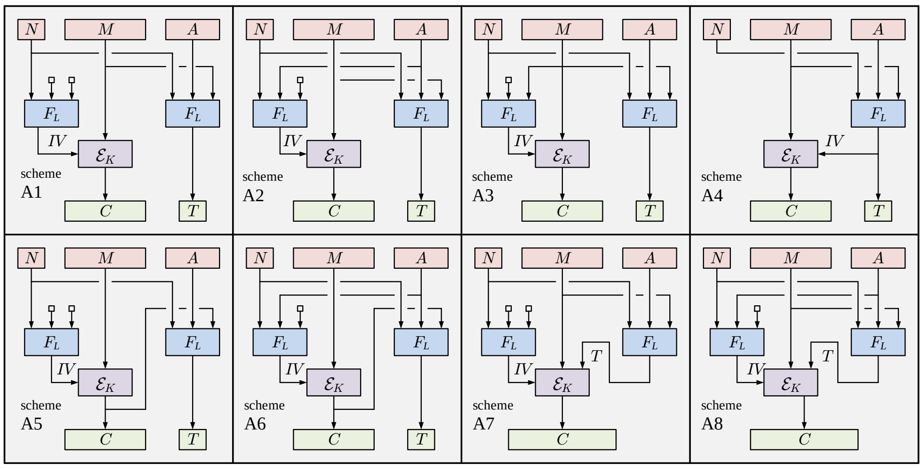

Fig. 2. The eight "favored" A-schemes. These convert an ivE scheme $\mathcal{E}$ and a vecMAC F into an nAE scheme. The IV is $F_L(N\,[, A]\,[, M])$ and the tag T is either $T = F_L(N, A, M)$ or $T = F_L(N, A, C)$. For this diagram we assume $F^{\mathrm{iv}} = F^{\mathrm{tag}} = F$.

more modest number of remaining schemes. The computer-assisted work was eventually rendered unnecessary by conventional proofs.

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 $\rightarrow$ 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 $\mathcal{E}_K(IV, M) = C \neq \perp$ implies that $\mathcal{D}_K(IV, C) = M$, while tidiness says that $\mathcal{D}_K(IV, C) = M \neq \perp$ 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 expect from deterministic encryption.

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. The conventional wisdom, that Encrypt-then-MAC is the only safe GC method, is arguably an artifact of having considered only pE + MAC $\rightarrow$ pAE conversions and admitting sloppy schemes.

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 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. 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 [2] pointing out that for the original nE definition of Rogaway [19], neither Encrypt-and-MAC nor MAC-then-Encrypt work for nE + MAC $\to$ 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.

A full version of the present paper is available [18]. It contains the proofs we have had to omit.

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.

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. Sampling from it is denoted $K \twoheadleftarrow \mathcal{K}$. 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

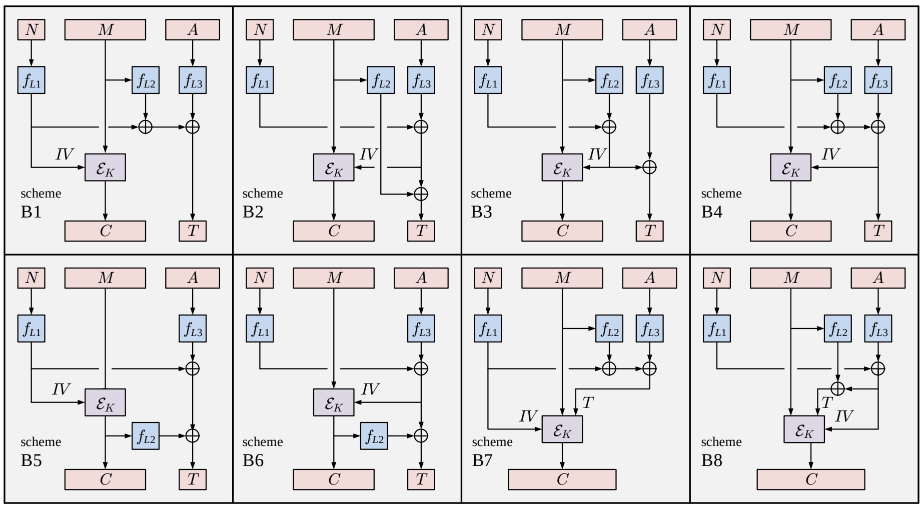

Fig. 3. The eight B-schemes corresponding to the favored A-schemes. Each converts an ivE scheme $\mathcal{E}$ and a strMAC f to an nAE scheme. The methods instantiate the vecMAC with a strMAC using the "XOR3" construction.

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 Fig. 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 Fig. 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 Fig. 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 nE scheme is *sloppy* if it satisfies everything but the tidiness condition. Might a "real world" nE 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. 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.

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 $\mathcal{X}$.

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

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

$\mathbf{Adv}_{\Pi}^{\mathrm{ivE}}(\mathcal{A}) = \Pr[\mathcal{A}^{\mathcal{E}(\cdot)} \Rightarrow 1] - \Pr[\mathcal{A}^{\$(\cdot)} \Rightarrow 1]$ 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 $IV \,\|\, \mathcal{E}_K(IV, M)$; and $\$(M)$ selects $IV \twoheadleftarrow \mathcal{IV}$, computes $C = \mathcal{E}_K(IV, M)$, returns $\perp$ if $C = \perp$, and otherwise returns $\big|IV \,\|\, C\big|$ random bits.

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

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

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 $F_K(\cdot)$. 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 $\mathcal{C} = C \,\|\, T$ is $\perp$; and if $IV = \perp$ then $C = \mathcal{E}_K(IV, M)$ is $\perp$.

3 AE from IV-Based Encryption and a vecMAC

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 $\text{ivE} + \text{strMAC} \to \text{nAE}$.

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 $F^{\text{iv}}\colon \mathcal{L} \times \mathcal{X}^{\text{iv}} \to \{0,1\}^\eta$, and a vecMAC $F^{\text{tag}}\colon \mathcal{L} \times \mathcal{X}^{\text{tag}} \to \{0,1\}^\tau$. Our constructions come in three types.

- **Type A$_1$ schemes**. The nAE scheme $\boldsymbol{\mathcal{E}} = \text{A}_1.\text{bbbbbb}[\mathcal{E}, F^{\text{iv}}, F^{\text{tag}}]$ defines $\boldsymbol{\mathcal{E}}^{N,A}_{K\,L}(M) = C \parallel T$ where $IV = F^{\text{iv}}_L(N \mid \sqcup,\ A \mid \sqcup,\ M \mid \sqcup)$, $C = \mathcal{E}_K(IV,\ M)$, and $T = F^{\text{tag}}_L(N \mid \sqcup,\ A \mid \sqcup,\ M \mid \sqcup)$. 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 $\text{bbbbbb} \in \{0,1\}^6$ 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 $\text{A}_1.100\,111[\mathcal{E}, F^{\text{iv}}, F^{\text{tag}}]$ sets $IV = F^{\text{iv}}_L(N, \sqcup, \sqcup)$ and $T = F^{\text{tag}}_L(N, A, M)$.
- **Type A$_2$ schemes**. The nAE scheme $\boldsymbol{\mathcal{E}} = \text{A}_2.\text{bbbbbb}[\mathcal{E}, F^{\text{iv}}, F^{\text{tag}}]$ defines $\boldsymbol{\mathcal{E}}^{N,A}_{K\,L}(M) = C \parallel T$ where $IV = F^{\text{iv}}_L(N \mid \sqcup,\ A \mid \sqcup,\ M \mid \sqcup)$, $C = \mathcal{E}_K(IV,\ M)$, and $T = F^{\text{tag}}_L(\ N \mid \sqcup,\ A \mid \sqcup,\ C)$. 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 $\boldsymbol{\mathcal{E}} = \text{A}_3.\text{bbbbbb}[\mathcal{E}, F^{\text{iv}}, F^{\text{tag}}]$ defines $\boldsymbol{\mathcal{E}}^{N,A}_{K\,L}(M) = C$ where $IV = F^{\text{iv}}_L(N \mid \sqcup,\ A \mid \sqcup,\ M \mid \sqcup)$, $T = F^{\text{tag}}_L(N \mid \sqcup,\ A \mid \sqcup,\ M \mid \sqcup)$, and $C = \mathcal{E}_K(IV, M \parallel T)$.

According to our conventions, the formulas above return $\boldsymbol{\mathcal{E}}^{N,A}_{K\,L}(M) = \perp$ 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 $(\mathcal{E}, F^{\text{iv}}, F^{\text{tag}})$, or it might be valid for some $(\mathcal{E}, F^{\text{iv}}, F^{\text{tag}})$ but not for others. We are only interested in candidate schemes $\boldsymbol{\mathcal{E}}$ with parameters $(\mathcal{E}, F^{\text{iv}}, F^{\text{tag}})$ that are *compatible*—ones where the specified composition does indeed satisfy the syntax of an nAE scheme. For example, with $\text{A}_1.001\,111$ (where $IV = F^{\text{iv}}_L(\sqcup, \sqcup, M)$ and $T = F^{\text{tag}}_L(N, A, M)$) there will never be a way to decrypt. And even for a scheme like $\text{A}_1.100\,111$ (where $IV = F^{\text{iv}}_L(N, \sqcup, \sqcup)$ and $T = F^{\text{tag}}_L(N, A, M)$), 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_L^0, F_L^1 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

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

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

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 ten provably secure A-schemes, nicknamed A1–A10. See Fig. 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 ten 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 bounds for A9 and A10 are inferior, due to the (somewhat curious) presence of ivE-advantage (ie, privacy) terms appearing in the authenticity bounds. 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–A10 and call them *favored*; the A9 and A10 schemes are 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: $T = F_L^{\text{tag}}(N, A, M)$ or $T = F_L(N, A, C)$.

Also shown in Fig. 5 are two *elusive* schemes, 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–A10. 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.) It may seem surprising that the security status of schemes

An	Scheme	IV	Tag	Sec	Comments
A1	$\mathrm{A}_1.100111$	$F_L^{\mathrm{iv}}(N,\sqcup,\sqcup)$	$F_L^{\mathrm{tag}}(N,A,M)$	yes	(Favored) C and T computable in parallel.
A2	$\mathrm{A}_1.110111$	$F_L^{\mathrm{iv}}(N,A,\sqcup)$	$F_L^{\mathrm{tag}}(N,A,M)$	yes	(Favored) C and T computable in parallel.
A3	$\mathrm{A}_1.101111$	$F_L^{\mathrm{iv}}(N,\sqcup,M)$	$F_L^{\mathrm{tag}}(N,A,M)$	yes	(Favored) Assume IV recoverable. Untruncatable.
A4	$\mathrm{A}_1.111111$	$F_L^{\mathrm{iv}}(N,A,M)$	$F_L^{\mathrm{tag}}(N,A,M)$	yes	(Favored) Assume $F^{\mathrm{iv}}=F^{\mathrm{tag}}$. Untruncatable. Nonce-reuse secure.
A5	$\mathrm{A}_2.100111$	$F_L^{\mathrm{iv}}(N,\sqcup,\sqcup)$	$F_L^{\mathrm{tag}}(N,A,C)$	yes	(Favored) Decrypt can validate T first, compute M and T in parallel.
A6	$\mathrm{A}_2.110111$	$F_L^{\mathrm{iv}}(N,A,\sqcup)$	$F_L^{\mathrm{tag}}(N,A,C)$	yes	(Favored) Decrypt can validate T first, compute M and T in parallel.
A7	$\mathrm{A}_3.100111$	$F_L^{\mathrm{iv}}(N,\sqcup,\sqcup)$	$F_L^{\mathrm{tag}}(N,A,M)$	yes	(Favored) Untruncatable.
A8	$\mathrm{A}_3.110111$	$F_L^{\mathrm{iv}}(N,A,\sqcup)$	$F_L^{\mathrm{tag}}(N,A,M)$	yes	(Favored) Untruncatable.
A9	$\mathrm{A}_3.110101$	$F_L^{\mathrm{iv}}(N,A,\sqcup)$	$F_L^{\mathrm{tag}}(N,\sqcup,M)$	yes	(Transitional)Weaker bound.Untruncatable.
A10	$\mathrm{A}_3.110011$	$F_L^{\mathrm{iv}}(N,A,\sqcup)$	$F_L^{\mathrm{tag}}(\sqcup,A,M)$	yes	(Transitional)Weaker bound.Untruncatable. XOR3 inapplicable.
A11	$\mathrm{A}_3.110001$	$F_L^{\mathrm{iv}}(N,A,\sqcup)$	$F_L^{\mathrm{tag}}(\sqcup,\sqcup,M)$	??	(Elusive) Security unresolved.
A12	$\mathrm{A}_3.100011$	$F_L^{\mathrm{iv}}(N,\sqcup,\sqcup)$	$F_L^{\mathrm{tag}}(\sqcup,A,M)$	??	(Elusive) Security unresolved.
—	all others	—	—	no	Counterexamples given.

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. Favored schemes were earlier pictured in Fig. 2.

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. In the full version [18], we discuss the technical challenges encountered, and also prove that A11 and A12 do provide authenticity under an additional security assumption, what we call the *knowledge-of-tags* assumption.

All A-schemes other than A1–A12 are insecure. In the full version of this paper we exhibit an attack on each of them [18]. 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 $F_K^{\mathrm{iv}}(\mathbf{x}) = F_K^{\mathrm{tag}}(\mathbf{x}) = F_K(\mathbf{x})$. 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.

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 the full version [18]. The characterization leaves a small "hole" (schemes A11 and A12); see the full version [18] for discussion and results about those two schemes. For compactness, our theorem statements are somewhat qualitative. But the proofs give a quantitative analysis of the reductions and concrete bounds.

Theorem 1 (Security of A1–10). Fix a compositional method $\mathrm{A}n \in \{\mathrm{A1}, \ldots, \mathrm{A10}\}$ and let $\mathcal{E} : \mathcal{K} \times \{0,1\}^{\eta} \times \mathcal{M} \to \{0,1\}^*$ be an ivE-scheme. Fix integers $1 \leq \eta, \tau \leq r$ and let $F \colon \mathcal{L} \times \mathcal{X} \to \{0,1\}^r$ be a vecMAC from which $F^{\mathrm{iv}} \colon \mathcal{L} \times \mathcal{X}^{\mathrm{iv}} \to \{0,1\}^{\eta}$ and $F^{\mathrm{tag}} \colon \mathcal{L} \times \mathcal{X}^{\mathrm{tag}} \to \{0,1\}^{\tau}$ are derived. Let the resulting nAE-scheme be denoted $\boldsymbol{\mathcal{E}} = \mathrm{A}n[\mathcal{E}, F^{\mathrm{iv}}, F^{\mathrm{tag}}]$. Then there are black-box 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 $1 \leq \eta, \tau \leq r$. 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 $F^{\mathrm{iv}} \colon \mathcal{L} \times \mathcal{X}^{\mathrm{iv}} \to \{0,1\}^{\eta}$ and $F^{\mathrm{tag}} \colon \mathcal{L} \times \mathcal{X}^{\mathrm{tag}} \to \{0,1\}^{\tau}$, 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.

4 AE from IV-Based Encryption and a strMAC

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 $\to$ 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 $F: \mathcal{L} \times (\mathcal{N} \times (\mathcal{A} \cup \{\sqcup\}) \times (\mathcal{M} \cup \{\sqcup\})) \to \{0,1\}^r$ from which functions $F^{\rm iv}$ and $F^{\rm tag}$ are defined. Here, we give a method to transform a strMAC $f: \mathcal{L} \times \mathcal{X} \to \{0,1\}^r$ into a vecMAC $F: \mathcal{L}^3 \times (\mathcal{X} \times (\mathcal{X} \cup \{\sqcup\}) \times (\mathcal{X} \cup \{\sqcup\})) \to \{0,1\}^r$, in order to instantiate A1–A9. We do this via the *three-xor construction*, defined by

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

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 $f: \mathcal{L} \times \mathcal{X} \to \{0,1\}^r$, 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 $\mathbf{Adv}^{\rm prf}_f(\mathcal{A}_f) \geq \mathbf{Adv}^{\rm prf}_F(\mathcal{A}_F)$. 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 $q_{\rm d}$, the number of decryption queries. But for the 1-decryption game, the sequence of adversarial queries is at-most-2-repeating (no repetitions among

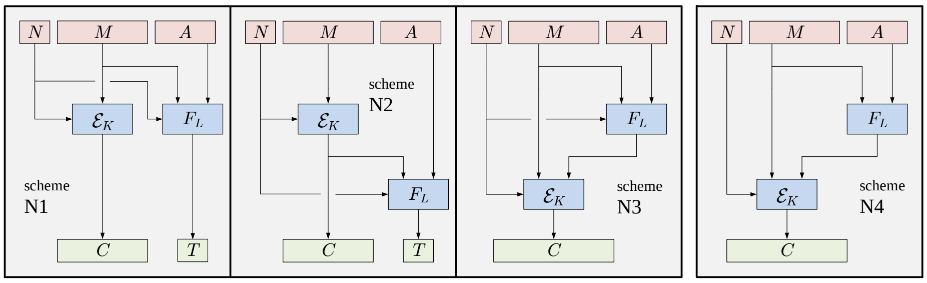

Fig. 6. Three correct N-schemes (left) and an elusive one (right). The methods achieve nE + vecMAC → nAE conversion. Application of the XOR3 construction to N1, N2, and N3 will result in three corresponding schemes that achieve nE+strMAC → nAE conversion. A second application of the XOR3 construction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

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 $q_{\rm d}$, which is due to the "weaker" definition for nAE security; we have not determined if this loss is artifactual or necessary. In Fig. 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 [7, 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 $\mathsf{c1}, \mathsf{c2}, \mathsf{c3}$.

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 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 $\mathcal{E} : \mathcal{K} \times \{0,1\}^\eta \times \mathcal{M} \to \{0,1\}^*$ 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 $\boldsymbol{\mathcal{E}}_{K\,L}^{N,A}(M) = C \,\|\, T$ where $C = \mathcal{E}_K(N, M)$ and $T = F_L(N \mid \sqcup,\ A \mid \sqcup,\ M \mid \sqcup)$.

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

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

- **Type $\mathbf{N}_2$ schemes**. The nAE scheme $\boldsymbol{\mathcal{E}} = \mathrm{N}_2.\mathrm{bbb}[\mathcal{E}, F]$ defines $\boldsymbol{\mathcal{E}}_{K\,L}^{N,A}(M) = C \parallel T$ where $C = \mathcal{E}_K(N,\ M)$ 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 $\mathbf{N}_3$ schemes.** The nAE scheme $\boldsymbol{\mathcal{E}} = \mathrm{N}_3.\mathrm{bbb}[\mathcal{E}, F]$ defines $\boldsymbol{\mathcal{E}}_{K\,L}^{N,A}(M) = C$ where $T = F_L(N \mid \sqcup,\ A \mid \sqcup,\ M \mid \sqcup)$ and $C = \mathcal{E}_K(N, M \parallel T)$.

As before, the formulas return $\boldsymbol{\mathcal{E}}_{K\,L}^{N,A}(M) = \perp$ 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 Fig. 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 Fig. 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 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 the full version. These proofs leave a small “hole,” which is scheme N4. For compactness, our theorem statements are again somewhat qualitative. But the proofs are not. They provide explicit reductions and quantitative analyses.

Theorem 3 (Security of N1–N3). Fix a compositional method $\mathrm{N}n \in \{\mathrm{N1}, \mathrm{N2}, \mathrm{N3}\}$ and integer $\tau \geq 1$. Fix an nE-scheme $\mathcal{E}\colon \mathcal{K} \times \{0,1\}^\eta \times \mathcal{M} \rightarrow \{0,1\}^*$ and a vecMAC $F\colon \mathcal{L} \times \mathcal{X} \rightarrow \{0,1\}^\tau$ that results in a valid nAE scheme $\boldsymbol{\mathcal{E}} = \mathrm{N}n[\mathcal{E}, F]$. 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 [2], 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.[1]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 ciphertext $C = C' \parallel T$ 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 Fig. 8. For example, let the adversary ask for the encryption of any message, obtaining a ciphertext $C = C' \parallel T$ 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.

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

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

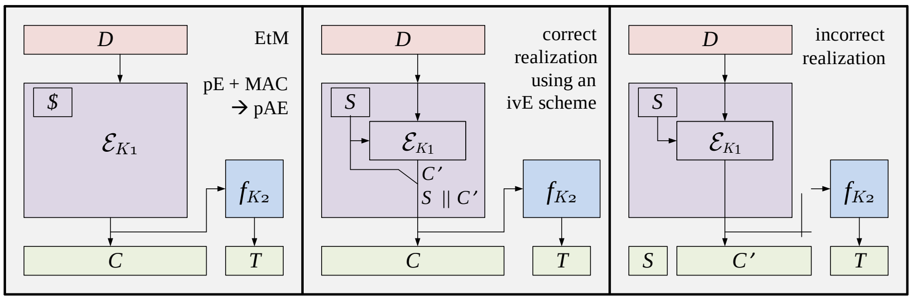

Fig. 8. Possible provenance of the ISO 19772 error. *Left*: The EtM method of BN, employing a probabilistic encryption algorithm $\mathcal{E}$ and a MAC f. The final ciphertext is $\mathcal{C} = C \parallel T$. *Middle*: A correct instantiation of EtM using an IV-based encryption scheme. With each encryption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

themselves, some ISO 9797 schemes are insecure when message lengths vary, a problem inherited by the enclosing AE scheme.

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. Much of this work was done while Rogaway was visiting Thammasat University during the summer of 2013. Many thanks to the NSF, who sponsored this work under NSF grants CNS 0904380, CNS 1228828, and CNS 1314885. Shrimpton also thanks the NSF, who sponsored his portion of this work under grants CNS 0845610 and CNS 1319061.

References

1. Alkassar, A., Geraldy, A., Pfitzmann, B., Sadeghi, A.R.: Optimized self-synchronizing mode of operation. In: Matsui, M. (ed.) FSE 2001. LNCS, vol. 2355,

pp. 78–91. Springer (Apr 2001)
2. Bellare, M., Tackmann, B.: Insecurity of MtE (and M&E) AEAD. Personal communications (unpublished note) (July 2013)
3. Bellare, M., Desai, A., Jokipii, E., Rogaway, P.: A concrete security treatment of symmetric encryption. In: 38th FOCS. pp. 394–403. IEEE Computer Society Press (Oct 1997)
4. Bellare, M., Namprempre, C.: Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. In: Okamoto, T. (ed.) ASIACRYPT 2000. LNCS, vol. 1976, pp. 531–545. Springer (Dec 2000)
5. Bellare, M., Namprempre, C.: Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. Journal of Cryptology 21(4), 469–491 (Oct 2008)
6. Bellare, M., Rogaway, P.: Encode-then-encipher encryption: How to exploit nonces or redundancy in plaintexts for efficient cryptography. In: Okamoto, T. (ed.) ASIACRYPT 2000. LNCS, vol. 1976, pp. 317–330. Springer (Dec 2000)
7. Bellare, M., Rogaway, P., Wagner, D.: The EAX mode of operation. In: Roy, B.K., Meier, W. (eds.) FSE 2004. LNCS, vol. 3017, pp. 389–407. Springer (Feb 2004)
8. Canvel, B., Hiltgen, A.P., Vaudenay, S., Vuagnoux, M.: Password interception in a SSL/TLS channel. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 583–599. Springer (Aug 2003)
9. Dworkin, M.: Recommendation for block cipher modes of operation: Methods and techniques. NIST Special Publication 800-38B (December 2001)
10. Dworkin, M.: Recommendation for block cipher modes of operation: The CCM mode for authentication and confidentiality. NIST Special Publication 800-38C (May 2004)
11. Dworkin, M.: 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. Goldwasser, S., Micali, S.: 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. Mitchell, C.: Personal communications (August 2011)
18. Namprempre, C., Rogaway, P., Shrimpton, T.: Reconsidering generic composition. Cryptology ePrint Archive, Report 2014/xxx (2014), full version of this paper
19. Rogaway, P.: Authenticated-encryption with associated-data. In: Atluri, V. (ed.) ACM CCS 02. pp. 98–107. ACM Press (Nov 2002)
20. Rogaway, P.: Nonce-based symmetric encryption. In: Roy, B.K., Meier, W. (eds.) FSE 2004. LNCS, vol. 3017, pp. 348–359. Springer (Feb 2004)
21. Rogaway, P., Shrimpton, T.: A provable-security treatment of the key-wrap problem. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 373–390. Springer (May / Jun 2006)
22. Vaudenay, S.: Security flaws induced by CBC padding — applications to SSL, IPSEC, WTLS,... In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 534–545. Springer (Apr / May 2002)