A Provable-Security Treatment of the Key-Wrap Problem

Phillip Rogaway[1] and Thomas Shrimpton[2]

[1] Dept. of Computer Science, University of California, Davis, California 95616, USA
[2] Dept. of Computer Science, Portland State University, Portland, Oregon 97201, USA

Abstract. We give a provable-security treatment for the *key-wrap problem*, providing definitions, constructions, and proofs. We suggest that key-wrap's goal is security in the sense of *deterministic authenticated-encryption* (DAE), a notion that we put forward. We also provide an alternative notion, a *pseudorandom injection* (PRI), which we prove to be equivalent. We provide a DAE construction, SIV, analyze its concrete security, develop a blockcipher-based instantiation of it, and suggest that the method makes a desirable alternative to the schemes of the X9.102 draft standard. The construction incorporates a method to turn a PRF that operates on a string into an equally efficient PRF that operates on a vector of strings, a problem of independent interest. Finally, we consider IV-based authenticated-encryption (AE) schemes that are maximally forgiving of repeated IVs, a goal we formalize as *misuse-resistant AE*. We show that a DAE scheme with a vector-valued header, such as SIV, directly realizes this goal.

1 Introduction

The American Standards Committee Working Group X9F1 has proposed four *key-wrap* schemes in a draft standard known as ANS X9.102, and NIST has promulgated a request for comments on the proposal [13]. The S/MIME working group of the IEEE had earlier adopted a key-wrap scheme [17], and their discussions on this topic go back to at least 1997 [36]. NIST is considering specifying a key-wrap mechanism in their own series of recommendations [M. Dworkin, personal communications]. But despite all this, the key-wrap goal would seem to be essentially unknown to the cryptographic community. No published paper analyzes any key-wrap scheme, and there is no formal definition for key wrap in the literature, let alone any proven-secure scheme. Consequently, the goal of this paper is to put the key-wrap problem on a proper, provable-security footing. In the process, we will learn quite a bit that's new about authenticated-encryption (AE).

Before proceeding it may be useful to give a very informal description of the key-wrap goal, echoing the wording in [13, p. 1]. A key-wrap scheme is a kind of shared-key encryption scheme. It aims to provide "privacy and integrity protection for specialized data such as cryptographic keys, ... without the use of nonces" (meaning counters or random bits). So key-wrap's raison d'être is to remove AE's reliance on a nonce or random bits. At least in the context of transporting cryptographic keys, a deterministic scheme should be just as good as a probabilistic one, anyway. Another goal of key wrap is to provide "integrity protection ... for cleartext associated data, ... which will typically contain control information about the wrapped key" [13, p. 1].

Contributions. We begin by offering a formal definition for what a key-wrap scheme should do, defining a goal we call *deterministic authenticated-encryption* (DAE). A thesis underlying our work is that the goal of a key-wrap scheme *is* DAE. In a DAE scheme, encryption deterministically turns a key, a header, and a message into a ciphertext. The header (which may be absent, a string, or even a vector of strings) is authenticated but not encrypted. To define security, the adversary is presented either a real encryption oracle and a real decryption oracle (both are deterministic), or else a bogus encryption oracle that just returns random bits and a bogus decryption oracle that always returns an indication of invalidity. For a good DAE scheme, the adversary should be unable to distinguish these possibilities. See Section 2.

Next we provide a DAE construction, SIV. (The acronym stands for *Synthetic IV*, where *IV* stands for *Initialization Vector*.) The construction combines a conventional IV-based encryption scheme (eg, CTR mode [27]) and a special kind of pseudorandom function (PRF)—one that takes a vector of strings as input. We prove that SIV is a good DAE, assuming its components are secure. See Section 3.

In practice one would want to realize SIV from a blockcipher, and so we show how to turn a PRF f that operates on a single string into a PRF f^* that takes a vector of strings. Under our S2V construction, the cost of computing the PRF $f^* = \mathrm{S2V}[f]$ on a vector $X = (X_1, \ldots, X_n)$ is at most the total cost to compute f on each component X_i, and it can be considerably less, as the contribution from a component X_i can be precomputed if it is to be held constant. See Section 4.

For a concrete alternative to the X9.102 schemes, we suggest to instantiate SIV using modes CTR and $\mathrm{CMAC}^* = \mathrm{S2V}[\mathrm{CMAC}]$, where CTR is counter mode [27] and CMAC is an arbitrary-input-length variant of the CBC MAC [28]. The specified mechanism removes unnecessary usage restrictions, improves efficiency, and provides provable security. See Section 5.

Applications of DAEs go beyond the wrapping of keys. Many IV-based encryption schemes, such as CBC, require an adversarially unpredictable IV. Experience has shown that implementers and protocol designers often supply an incorrect IV, such as a constant or counter. In a *misuse-resistant* AE scheme the aim is to do as well as possible with whatever IV is provided. We formalize this goal and show that a DAE scheme that takes a vector-valued header provides an immediate solution: just regard the IV as one component of the header. Adopting this viewpoint, SIV can be regarded as an IV-based AE scheme, one as efficient with respect to blockcipher calls as conventional two-pass AE schemes like CCM [29] but more resilient to IV misuse. See Section 6.

Finally, we investigate the basic properties of DAEs. First, we give an alternative characterization of DAEs. A *pseudorandom injection* (PRI) is like a blockcipher except that the ciphertext may be longer than the plaintext (also, the message space may be richer than $\{0,1\}^n$ for some fixed n, and a header may be provided). We prove PRIs equivalent to DAEs, up to a term that is negligible when the PRI is adequately length-increasing. Next, we explain that the "all-in-one" definition we adopt for DAEs is equivalent to a more conventional, two-requirement (privacy-plus-authenticity) definition. Finally, we sketch a result validating the intuition that DAE-encrypting a message that includes a random key provides semantic security. See Section 7.

Why this goal? There are two main reasons to prefer DAE over conventional (probabilistic or stateful) AE. First, DAE saves one from having to introduce random bits or state in contexts where these measures are infeasible or unnecessary. Relatedly, DAE saves on bandwidth, since no nonce or random value need be sent.

That said, in many contexts where one would think to use key wrap, one *can* use a conventional AE scheme, instead. This does not make studying the key-wrap problem pointless. First, it clarifies the relationship between key wrap and conventional AE. Second, DAE leads to misuse-resistant AE, and methods that achieve this aim make practical alternatives to conventional (not misuse-resistant) two-pass AE methods. Finally, practitioners have already "voted" for key-wrap by way of protocol-design and standardization efforts, and it is simply not productive to say "use a conventional AE scheme" after this option has been rejected.

Further related work. AE goals were formalized over a series of papers [6, 8, 20, 31, 33]. The idea of binding the encryption process to unencrypted strings is folklore, with recent work in this direction including [23, 31, 35]. Russell and Wong [34] introduce a completely different approach for dealing with the encryption of low-entropy messages, and Dodis and Smith [12] extend this entropy-based approach. Phan and Pointcheval [30] study relationships among security notions for conventional (length-preserving and headerless) ciphers. The SIV construction resembles the AE scheme EAX [9]. A less ambitious relaxation on IV requirements than that formalized as misuse-resistant encryption is given in [32]. A full version of this paper is available from the authors' web pages.

2 DAE Security

Notation. For a distribution $\mathcal{S}$ let $S \xleftarrow{\$} \mathcal{S}$ mean that S is selected randomly from $\mathcal{S}$ (if $\mathcal{S}$ is a finite set the assumed distribution is uniform). All strings are binary strings. When X and Y are strings we write $X \| Y$ for their concatenation. When $X \in \{0,1\}^*$ is a string $|X|$ is its length and, if $1 \le i \le j \le |X|$, then $X[i..j]$ is the substring running from its i^{th} to j^{th} characters, or the empty string ε otherwise. By a vector we mean a sequence of zero or more strings, and we write $\{0,1\}^{**}$ for the space of all vectors. We write a vector as $X = (X_1, \ldots, X_n)$ where

$n = |X|$ is its number of components. If $X = (X_1, \ldots, X_n)$ and $Y = (Y_1, \ldots, Y_m)$ are vectors then X, Y is the vector $(X_1, \ldots, X_n, Y_1, \ldots, Y_m)$. In pseudocode, Boolean variables are silently initialized to `false`, sets are initialized to the empty set, and partial functions are initialized to everywhere undefined (set to `undef`). An *adversary* is an algorithm with access to one or more oracles, which we write as superscripts. By $A^{\mathcal{O}} \Rightarrow 1$ we mean the event that adversary A, running with its oracle $\mathcal{O}$, outputs 1. When an adversary has an oracle with an expressed domain D we understand that the oracle returns the distinguished value $\perp$, read as *invalid*, if the adversary asks a query outside of D.

SYNTAX. A scheme for *deterministic authenticated-encryption*, or DAE, is a tuple $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$. The *key space* $\mathcal{K}$ is a set of strings or infinite strings endowed with a distribution. For a practical scheme there must be a probabilistic algorithm that samples from $\mathcal{K}$, and we identify this algorithm with the distribution it induces. The *encryption algorithm* $\mathcal{E}$ and *decryption algorithm* $\mathcal{D}$ are deterministic algorithms that take an input in $\mathcal{K} \times \{0,1\}^{**} \times \{0,1\}^*$ and return either a string or the distinguished value $\perp$. We write $\mathcal{E}_K^H(X)$ or $\mathcal{E}_K(H, X)$ for $\mathcal{E}(K, H, X)$ and $\mathcal{D}_K^H(Y)$ or $\mathcal{D}_K(H, Y)$ for $\mathcal{D}(K, H, Y)$. We assume there are sets $\mathcal{H} \subseteq \{0,1\}^{**}$, the *header space*, and $\mathcal{X} \subseteq \{0,1\}^*$, the *message space*, such that $\mathcal{E}_K^H(X) \in \{0,1\}^*$ iff $H \in \mathcal{H}$ and $X \in \mathcal{X}$. We assume that $X \in \mathcal{X} \Rightarrow \{0,1\}^{|X|} \subseteq \mathcal{X}$. The *ciphertext space* is $\mathcal{Y} = \{\mathcal{E}_K^H(X) \colon K \in \mathcal{K},\ H \in \mathcal{H},\ X \in \mathcal{X}\}$. We require $\mathcal{D}_K^H(Y) = X$ if $\mathcal{E}_K^H(X) = Y$, and $\mathcal{D}_K^H(Y) = \perp$ if there is no such X. It will be our convention that $\mathcal{E}_K^H(\perp) = \mathcal{D}_K^H(\perp) = \perp$ for all $K \in \mathcal{K}$ and $H \in \mathcal{H}$. For any $K \in \mathcal{K}$, $H \in \mathcal{H}$, and $X \in \mathcal{X}$, we assume that $|\mathcal{E}_K^H(X)| = |X| + e(H, X)$ for a function $e \colon \{0,1\}^{**} \times \{0,1\}^* \to \mathbb{N}$ where $e(H, X)$ depends only on the number of components of H, the length of each of these components, and the length of X. The function e is called the *expansion function* of the DAE scheme. Often we are concerned with the minimum expansion that might arise, and so define the number $s = \min_{H \in \mathcal{H}, X \in \mathcal{X}} \{e(H, X)\}$ as the *stretch* of the scheme.

Among what is formalized above: (1) encryption and decryption are given by algorithms, not just functions; (2) trying to encrypt something outside of the header space or message space returns $\perp$; (3) trying to decrypt something that isn't the encryption of anything returns $\perp$; (4) if you can encrypt a string of some length you can encrypt all strings of that length; and (5) the length of a ciphertext exceeds the length of the plaintext by an amount that depends on, at most, the length of the plaintext and the length of the components of the header.

A DAE is *length-preserving* if $e(H, X) = 0$ for all $H \in \mathcal{H}$, $X \in \mathcal{X}$. An *enciphering scheme* is a length-preserving DAE. A *tweakable blockcipher* is an enciphering scheme where the plaintext space is $\mathcal{X} = \{0,1\}^n$ for some $n \geq 1$. A *blockcipher* is a tweakable blockcipher where the header space $\mathcal{H} = \{\varepsilon\}$ is a singleton set; as such, we omit mention of it and write $E \colon \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$.

SECURITY. We now give our formalization for DAE security.

Definition 1. *Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be a DAE scheme with header space $\mathcal{H}$, message space $\mathcal{X}$, and expansion function e. The* ***DAE-advantage*** *of adversary A*

in breaking Π is defined as

$$\mathbf{Adv}_{\Pi}^{\mathrm{dae}}(A) = \Pr\left[K \xleftarrow{\$} \mathcal{K}:\ A^{\mathcal{E}_K(\cdot,\cdot),\ \mathcal{D}_K(\cdot,\cdot)} \Rightarrow 1\right] - \Pr\left[A^{\$(\cdot,\cdot),\ \perp(\cdot,\cdot)} \Rightarrow 1\right]. \quad \blacksquare$$

On query $H \in \mathcal{H}$, $X \in \mathcal{X}$, the adversary's *random-bits* oracle $\$(\cdot,\cdot)$ returns a random string of length $|X| + e(H,X)$. As always, oracle queries outside the specified domain return $\perp$. The $\perp(\cdot,\cdot)$ oracle returns $\perp$ on every input. We assume that the adversary does not ask (H,Y) of its right (ie, second) oracle if some previous left (ie, first) oracle query (H,X) returned Y; does not ask (H,X) of its left oracle if some previous right-oracle query (H,Y) returned X; does not ask left queries outside of $\mathcal{H} \times \mathcal{X}$; and does not repeat a query. The last two assumptions are without loss of generality, as an adversary that violated any of these constraints could be replaced by a more efficient and equally effective adversary (in the $\mathbf{Adv}_{\Pi}^{\mathrm{dae}}$-sense) that did not. The first two assumptions are to prevent trivial wins.

Discussion. The DAE-notion of security directly captures the amalgamation of privacy and authenticity. Assume that $\mathbf{Adv}_{\Pi}^{\mathrm{dae}}(A)$ is insignificantly small for any reasonable adversary. Then, for privacy, we know that any sequence of distinct $\mathcal{E}_K$-queries results in a distribution on outputs resembling a distribution on outputs that depends only on the length of each query (in fact, the outputs look like random strings of the appropriate lengths). For authenticity we have that, despite the ability to perform a chosen-plaintext attack (as provided by the $\mathcal{E}_K$ oracle), we are unable to come up with a new query Y for which $\mathcal{D}_K^H(Y) \neq \perp$.

It is possible to disentangle the privacy and authenticity notions in the DAE definition, defining separate notions for deterministic privacy and deterministic authenticity. While the traditional approach for defining AE has been to split the goal into two separate properties, the unified definition seems to us nicer and more succinct.

We point out that the DAE notion does not formalize the idea that the party that produces a valid ciphertext (a value that decrypts to something other than $\perp$) necessarily *knows* the underlying key K. One could formalize this, but it would not coincide with DAE. Sometimes the key-wrap goal has been described in these terms. We suspect that when security-designers speak of having to know the key in order to produce a valid ciphertext what they typically mean is not a proof of knowledge, but just the inability for a party to produce a valid ciphertext in the absence of the key. It is the latter notion that is well captured by our DAE definition.

3 Building a DAE Scheme: The SIV Construction

Conventional IV-based encryption schemes. Encryption modes like CBC and CTR are what we call *conventional* IV-based encryption schemes. Such a scheme $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ is syntactically similar to a DAE but in this context the header space $\mathcal{H}$ is a set of strings and is renamed the *IV space*, $\mathcal{IV}$. We expect only privacy in a conventional IV-based encryption scheme, and demand a random IV. This makes the security notion rather weak, but sufficient for our purposes. The following definition captures the desired notion.

Fix a conventional IV-based encryption scheme $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ with IV-space $\mathcal{IV} = \{0,1\}^n$. For simplicity, assume Π is length-preserving. Let $\mathcal{E}^\$$ be the probabilistic algorithm defined from $\mathcal{E}$ that, on input $K \in \mathcal{K}$ and $M \in \{0,1\}^*$, chooses an $IV \xleftarrow{\$} \{0,1\}^n$, computes $C \leftarrow \mathcal{E}_K^{IV}(M)$ and returns $IV \parallel C$. Then we define the advantage of adversary A in violating the privacy of Π by

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

where the $\$(\cdot)$ oracle, on input M, returns a random string of length $n+|M|$. We assume that the adversary never asks a query M outside of the message space $\mathcal{X}$ of Π.

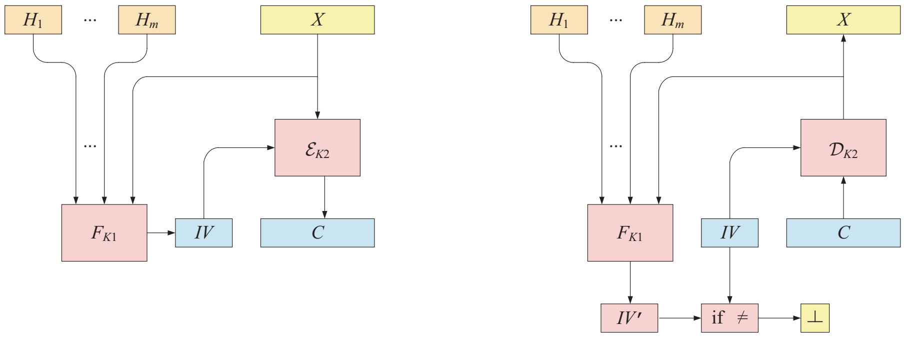
Arbitrary-input pseudorandom functions. Fix nonempty sets $\mathcal{K}$ and $\mathcal{X}$, the first being finite or otherwise endowed with a distribution and the second being finite or countably infinite. A *pseudorandom function* (PRF) is a map $F\colon \mathcal{K} \times \mathcal{X} \to \{0,1\}^n$ for some $n \geq 1$. We write $F_K(X)$ for $F(K,X)$. Let $\mathrm{Func}(\mathcal{X}, \mathcal{Y})$ be the set of all functions from $\mathcal{X}$ to $\mathcal{Y}$ and let $\mathrm{Func}(\mathcal{X}, n) = \mathrm{Func}(\mathcal{X}, \{0,1\}^n)$. Regarding a function as the key, we can consider $\mathrm{Func}(\mathcal{X}, n)$ to be a PRF; to each $X \in \mathcal{X}$ associate a random string in $\{0,1\}^n$. Let A be an adversary. The advantage of A in violating the pseudorandomness of F is

$$\mathbf{Adv}_F^{\mathrm{prf}}(A) = \Pr\left[K \leftarrow \mathcal{K} :\ A^{F_K(\cdot)} \Rightarrow 1\right] - \Pr\left[\rho \xleftarrow{\$} \mathrm{Func}(\mathcal{X}, n) :\ A^{\rho(\cdot)} \Rightarrow 1\right] .$$

It is tacitly assumed that the adversary has a mechanism of naming points in $\mathcal{X}$ by strings; if $\mathcal{X} \subseteq \{0,1\}^*$ then a string names itself, but if $\mathcal{X}$ is not a set of strings then points of $\mathcal{X}$ are encoded as strings in some natural way. Our definition of PRFs is unusual for allowing the input X to be arbitrary (possibly not a string).

The SIV construction. Let $F\colon \mathcal{K}_1 \times \{0,1\}^{**} \to \{0,1\}^n$ be a PRF. Let $\Pi = (\mathcal{K}_2, \mathcal{E}, \mathcal{D})$ be a conventional IV-based encryption scheme with IV-length n and message space $\mathcal{X}$. We write $F_K(H,M)$ instead of $F_K((H,M))$. We construct from (F, Π) a DAE $\widetilde{\Pi} = \mathrm{SIV}[F,\Pi] = (\widetilde{\mathcal{K}}, \widetilde{\mathcal{E}}, \widetilde{\mathcal{D}})$ with header space $\{0,1\}^{**}$ and message space $\mathcal{X}$ where $\widetilde{\mathcal{K}} = \mathcal{K}_1 \times \mathcal{K}_2$ and the encryption and decryption algorithms are as illustrated and defined in Fig. 1. Recall that $Y[n+1..|Y|] = \varepsilon$ if $|Y| < n$.

We will now show that if F is PRF-secure and Π is IND\$-secure then $\widetilde{\Pi} = \mathrm{SIV}[F,\Pi]$ is DAE-secure. The intuition behind the proof is this. If any bit of the header H or plaintext X is new then the string IV will look like a random string and so $IV \parallel C$ will be difficult to distinguish from random bits. On decryption, the adversary must create a new (H,Y) where $Y = IV \parallel C$. Let's imagine giving the adversary the corresponding plaintext X for free. Now (H,X) is new because (H,X) determines (H,Y) and the adversary is not allowed to decipher values that it trivially knows the decipherment of. But if (H,X) is new then IV' is adversarially unpredictable and so its chance of being equal to IV is only about 2^{-n}.



Algorithm $\widetilde{\mathcal{E}}_{K1,K2}(H,X)$
$IV \leftarrow F_{K1}(H,X)$
$C \leftarrow \mathcal{E}_{K2}^{IV}(X)$
return $Y \leftarrow IV \parallel C$

Algorithm $\widetilde{\mathcal{D}}_{K1,K2}(H,Y)$
if $|Y| < n$ **then return** $\perp$
$IV \leftarrow Y[1\,..\,n],\quad C \leftarrow Y[n+1\,..\,|Y|]$
$X \leftarrow \mathcal{D}_{K2}^{IV}(C)$
$IV' \leftarrow F_{K1}(H,X)$
if $IV = IV'$ **then return** X **else return** $\perp$

Fig. 1. The SIV construction. The left side illustrates and defines encryption, the right side, decryption. The header is $H = (H_1, \ldots, H_m)$, the plaintext is X, the key is $(K1, K2)$, and the ciphertext is $Y = IV \parallel C$. Function $F\colon \mathcal{K}_1 \times \{0,1\}^{**} \to \{0,1\}^n$ is a PRF and $(\mathcal{K}_2, \mathcal{E}, \mathcal{D})$ is an IV-based encryption scheme, such as CTR mode.

In the following result we write $\mathrm{Time}_{\Pi}(\mu)$, where $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ is an IV-based encryption scheme and $\mu > 0$ is an integer, for the sum of the worst-case times: to select $K \stackrel{\$}{\leftarrow} \mathcal{K}$, to compute $\mathcal{E}_K^{IV}$ on inputs of total length μ, and to compute $\mathcal{D}_K^{IV}$ on inputs of total length μ. Here, by convention, "time" means actual running time plus program size, all relative to some fixed RAM model of computation.

Theorem 1. *Let $F\colon \mathcal{K}_1 \times \{0,1\}^{**} \to \{0,1\}^n$ be a PRF and let $\Pi = (\mathcal{K}_2, \mathcal{E}, \mathcal{D})$ be a conventional IV-based encryption scheme with message space $\mathcal{X}$ and IV-length n. Let $\widetilde{\Pi} = \mathrm{SIV}[F, \Pi]$. Let A be an adversary (for attacking $\widetilde{\Pi}$) that runs in time t and asks q queries, these of total length μ. Then there exists adversaries B and D such that*

$$\mathbf{Adv}_{\Pi}^{\mathrm{priv\$}}(B) + \mathbf{Adv}_{F}^{\mathrm{prf}}(D) \geq \mathbf{Adv}_{\widetilde{\Pi}}^{\mathrm{dae}}(A) - q/2^n \,.$$

What is more, B and D run in time at most $t' = t + \mathrm{Time}_{\Pi}(\mu) + c\mu$ for some absolute constant c and ask at most q queries, these of total length μ. ∎

Proof. The proof proceeds in two stages. First we consider the DAE scheme $G = \mathrm{SIV}[\mathrm{Func}(\{0,1\}^{**}, n), \Pi]$ (replacing the function F_{K1} with a random function $\rho \in \mathrm{Func}(\{0,1\}^{**}, n)$). Then we extend this to account for the insecurity of the PRF F.

Denote the forward and reverse algorithms associated to G as $G_{\rho,K2}$ and $G^{-1}_{\rho,K2}$, with $(\rho, K2)$ being the key. Let $\delta = \mathbf{Adv}^{\mathrm{dae}}_{G}(A)$ and $q = q_{\mathrm{L}} + q_{\mathrm{R}}$ and $\mu = \mu_{\mathrm{L}} + \mu_{\mathrm{R}}$ where q_{L} and q_{R} are the number of left and right oracle queries, these totaling μ_{L} and μ_{R} bits, respectively. With the obvious simplifications in notation we have

$$\begin{aligned}
\delta &= \Pr\left[A^{G_{\rho,K2}(\cdot,\cdot),\, G^{-1}_{\rho,K2}(\cdot,\cdot)} \Rightarrow 1\right] - \Pr\left[A^{\$(\cdot,\cdot),\, \perp(\cdot,\cdot)} \Rightarrow 1\right] \\
&= \left(\Pr\left[A^{G_{\rho,K2}(\cdot,\cdot),\, G^{-1}_{\rho,K2}(\cdot,\cdot)} \Rightarrow 1\right] - \Pr\left[A^{G_{\rho,K2}(\cdot,\cdot),\, \perp(\cdot,\cdot)} \Rightarrow 1\right]\right) \\
&\quad + \left(\Pr\left[A^{G_{\rho,K2}(\cdot,\cdot),\, \perp(\cdot,\cdot)} \Rightarrow 1\right] - \Pr\left[A^{\$(\cdot,\cdot),\, \perp(\cdot,\cdot)} \Rightarrow 1\right]\right) \;=\; p_1 + p_2
\end{aligned}$$

where p_1 and p_2 represent the corresponding parenthesized expressions; it remains to bound these quantities. For p_2 we construct from A an adversary B^g for attacking the priv\$-security of Π. Let B run A. When A asks its left-oracle a query (H, X), let B ask $g(M)$ and return the result to A. When A asks a right-oracle query have B return $\perp$. When A halts with output bit b, let B output b. Notice that if $g = \mathcal{E}^{\$}_{K}$ then B properly simulates $G_{\rho,K2}(\cdot,\cdot), \perp(\cdot,\cdot)$ oracles for A (here we need the assumption that A never repeats a query). Similarly, if $g = \$$ then B simulates $\$(\cdot,\cdot), \perp(\cdot,\cdot)$ oracles for A. Hence $p_2 \le \mathbf{Adv}^{\mathrm{priv\$}}_{\Pi}(B)$.

To bound p_1 consider giving the key $K2$ to the adversary and then asking it to carry out its distinguishing task. As this can only make the task easier we may assume

$$\begin{aligned}
p_1 &= \Pr\left[A^{G_{\rho,K2}(\cdot,\cdot),\, G^{-1}_{\rho,K2}(\cdot,\cdot)} \Rightarrow 1\right] - \Pr\left[A^{G_{\rho,K2}(\cdot,\cdot),\, \perp(\cdot,\cdot)} \Rightarrow 1\right] \\
&\le \Pr\left[A(K2)^{G_{\rho,K2}(\cdot,\cdot),\, G^{-1}_{\rho,K2}(\cdot,\cdot)} \Rightarrow 1\right] - \Pr\left[A(K2)^{G_{\rho,K2}(\cdot,\cdot),\, \perp(\cdot,\cdot)} \Rightarrow 1\right] .
\end{aligned}$$

We can assume without loss of generality that A halts and outputs 1 as soon as a right-oracle query returns something other than $\perp$. Under this assumption, encryption queries are useless for distinguishing between these two oracle pairs, as prior to the right oracle returning $M \ne \perp$ both pairs behave as $G_{\rho,K2}(\cdot,\cdot), \perp(\cdot,\cdot)$. Hence p_1 is bounded by the probability that A asks a right-oracle query (H, Y) such that $G^{-1}_{\rho,K2}(H, Y) \ne \perp$. Examining the algorithm for $G^{-1}_{\rho,K2}$ we see that this occurs only when $\rho(H, X) = IV$, where $X = \mathcal{D}^{IV}_{K2}(C)$ (with Y having been parsed into IV and C). Since the adversary is given the key $K2$, it can compute $\mathcal{D}^{IV}_{K2}(C)$ for any strings IV, C of its choosing. In particular, when it asks a right-oracle query (H, Y) it knows what is the input to the random function ρ and what is the target output IV. But under our assumption that A never queries its right oracle (H, Y) when some left-oracle query (H, X) returned Y, either the input (H, X) is new, or the target IV is new. Thus, the probability that $\rho(H, X) = IV$ is at most $1/2^n$ for each right-oracle query, and we conclude that $p_1 \le q_{\mathrm{R}}/2^n$. Since $q_{\mathrm{R}} \le q$ we have $\delta \le \mathbf{Adv}^{\mathrm{priv\$}}_{\Pi}(B) + q/2^n$.

For the second part of the proof note that

$$\mathbf{Adv}^{\mathrm{dae}}_{\widetilde{\Pi}}(A) = \delta + \Pr\left[A^{\widetilde{\mathcal{E}}_{K1,K2}(\cdot,\cdot), \widetilde{\mathcal{D}}_{K1,K2}(\cdot,\cdot)} \Rightarrow 1\right] - \Pr\left[A^{G_{\rho,K2}(\cdot,\cdot), G^{-1}_{\rho,K2}} \Rightarrow 1\right]$$

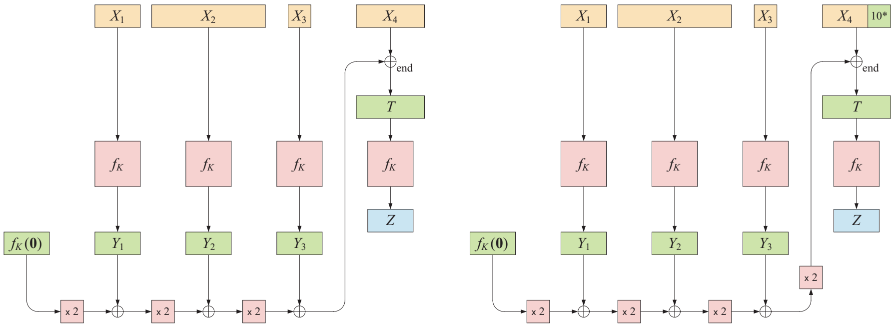
where $\widetilde{\Pi} = (\mathcal{K}1 \times \mathcal{K}2, \widetilde{\mathcal{E}}, \widetilde{\mathcal{D}})$ and we have suppressed the random selections $K1 \xleftarrow{\$} \mathcal{K}_1$ and $K2 \xleftarrow{\$} \mathcal{K}_2$. Let D^g be an adversary for attacking F as a PRF, and let it operate as follows. Adversary D picks $K2 \xleftarrow{\$} \mathcal{K}_2$ and runs A. When A asks a left oracle query (H, X), B answers by setting $IV \leftarrow g(H, X)$, computing $C \leftarrow \mathcal{E}_{K2}^{IV}(X)$ and returning to A the string $IV \,||\, C$. On a right oracle query (H, Y), adversary D parses $IV = Y[1..n]$, $C = Y[n+1..|Y|]$, computes $X \leftarrow \mathcal{D}_{K2}^{IV}(C)$ and tests if $IV = g((H, X))$, returning X to A if so and $\perp$ otherwise. When A halts with output bit b, let D output b. Clearly D correctly simulates $\widetilde{\mathcal{E}}_{K1,K2}(\cdot,\cdot), \widetilde{\mathcal{D}}_{K1,K2}(\cdot,\cdot)$ when its oracle $g = F_{K1}$ for some random key $K1$, and $G_{K1,K2}(\cdot,\cdot), G_{K1,K2}^{-1}(\cdot,\cdot)$ if instead $g = \rho$ for a random $\rho \in \mathrm{Func}(\mathcal{M}, n)$. So, $\mathbf{Adv}_{\widetilde{\mathcal{E}}}^{\mathrm{dae}}(A) \leq \delta + \mathbf{Adv}_F^{\mathrm{prf}}(D)$ and rearranging gives the result.

4 Enriching a PRF to take Vectors of Strings as Input

The goal. Traditionally, a pseudorandom function (PRF) takes a single string as input: under the control of a key K, a PRF f maps a string $X \in \{0,1\}^*$ into a string $f_K(X)$. But SIV uses a non-traditional PRF—a function F that, under the control of a key K, maps a vector of strings $X = (X_1, \ldots, X_m) \in \{0,1\}^{**}$ into a string $F_K(X)$. Let us call a PRF that takes a string as input an sPRF (string-input PRF) and a PRF that takes a vector of strings as input a vPRF (vector-input PRF). This section is about efficient ways to turn an sPRF f into a vPRF f^*.

At first glance it might seem like there'd be little to say about sPRF-to-vPRF conversion: there's an obvious approach for solving the problem, and it's obviously correct. Namely, encode any vector of strings $X = (X_1, \ldots, X_m)$ into a single string $\langle X \rangle$ and apply the sPRF to that, $f_K^*(X) = f_K(\langle X \rangle)$. By *encode* we mean any reversible, easily-computed map of a vector of strings into a single one, say $\langle X_1, \ldots, X_m \rangle = X_1 \| N_1 \| \cdots \| X_m \| N_m$ where $N_i = |X_i|_{64}$ is the length of X_i encoded into 64 bits (assume that $|X_i| < 2^{64}$ for all i). The problem with making a vPRF in such a way is a diminution of efficiency. First, computing $f_K^*(X)$ may take longer than the total time to compute $f_K(X_i)$ for each component X_i since we have added $64m$ bits for length annotation. Second, even if some components of X stay fixed (say X_2 is constant), we must still re-process the entire encoded string each time we compute f_K^* at a new value. Third, the mechanism is not parallelizable; one cannot process X_i until one is done processing X_{i-1}. Fourth, the assumption that $|X_i| < 2^{64}$, while reasonable in practice, is artificial and potentially wasteful, yet use of a stingier encoding will lead to greater complexity. Finally, the given encoding disrupts word alignment: if, for example, the first argument is one byte and all subsequent arguments are multiples of eight bytes, an implementation will now be dealing with non-word-aligned data. Fixing this problem by a smarter encoding will lead to increased complexity. We aim to do sPRF-to-vPRF conversion in a way that fixes the problems above.

Notation. Fix a value $n \geq 2$. Let $\mathbf{0} = 0^n$ and $\mathbf{1} = 0^{n-1}1$ and $\mathbf{2} = 0^{n-2}10$. These are regarded as points in finite field $\mathbb{F}_{2^n}$ represented using a primitive polynomial



Algorithm $f^*_K(X_1, \ldots, X_m)$ *The S2V Construction,* $f^* = \mathrm{S2V}[f]$

if $m = 0$ **then return** $f_K(\mathbf{1})$
$S \leftarrow f_K(\mathbf{0})$
for $i \leftarrow 1$ **to** $m - 1$ **do** $S \leftarrow \mathbf{2}S \oplus f_K(X_i)$
if $|X_m| < n$ **then** $T \leftarrow S \oplus_{\mathrm{end}} X_m$ **else** $T \leftarrow \mathbf{2}S \oplus_{\mathrm{end}} X_m 10^*$
return $Z \leftarrow f_K(T)$

Fig. 2. The S2V construction makes a PRF $f^*\colon \mathcal{K} \times \{0,1\}^{**} \to \{0,1\}^n$ from a PRF $f\colon \mathcal{K} \times \{0,1\}^* \to \{0,1\}^n$. **Bottom:** Definition of S2V. Strings $X_1, \ldots, X_m \in \{0,1\}^*$ and $m \geq 0$ are arbitrary. **Top:** Illustration of it, computing $Z = f^*_K(X_1, X_2, X_3, X_4)$. The left side shows the case when $|X_4|$ is a nonzero multiple of n bits, the right otherwise.

in the customary way. For $S \in \{0,1\}^n$ let $\mathbf{2}S$ mean the n-bit string representing the product of $\mathbf{2}$ and S. This can be computed with a left shift of S followed by a conditional xor. By $\mathbf{2}^i S$ we mean to do this multiplication by $\mathbf{2}$ a total of i times. By $N \oplus_{\mathrm{end}} X$ ("xor-into-the-end") we mean to xor the n-bit string N into the end of the string X, which will have at least n bits; $N \oplus_{\mathrm{end}} X = (0^{x-n}N) \oplus X$ where $x = |X|$. By $X10^*$ we mean $X10^i$ where $i \geq 0$ is the least number such that $|X| + 1 + i$ is divisible by n.

The S2V construction. Let $f\colon \mathcal{K} \times \{0,1\}^* \to \{0,1\}^n$ be an sPRF. We construct from it the vPRF $f^* = \mathrm{S2V}[f]$ where $f^*\colon \mathcal{K} \times \{0,1\}^{**} \to \{0,1\}^n$ is specified and illustrated in Fig. 2. The special treatment of the last component of input, X_m, is to handle the case where $|X_m| < n$. The construction has the desired efficiency characteristics. The time to compute $f^*_K(X)$ is essentially the sum of the times to compute $f_K(X_i)$ on each component; in particular, when $f = \mathrm{CMAC}$, say, the number of blockcipher calls to compute $f^*_K(X)$ is the sum of the number of blockcipher calls to compute each $f_K(X_i)$. Also, one can preprocess invariant components so that the time to compute $f^*_K(X)$ will not significantly depend on them. The computation of f^* is on-line (assuming that f itself is on-line); in particular, the component lengths need not be known in advance. Word alignment is not disrupted. And the scheme is parallelizable: different arguments can be acted on simultaneously, so f^* will be parallelizable if f is.

In a related effort we have proven the following result. The complexity-theoretic analog of Theorem 2 follows in the usual way.

Theorem 2. *Let $f = \mathrm{Func}(\{0,1\}^*, n)$ and $f^* = \mathrm{S2V}[f]$. Let A be an adversary that asks at most $q \geq 3$ vector-valued queries having p components in all, and each vector having fewer than n components. Then $\mathbf{Adv}^{\mathrm{prf}}_{f^*}(A) \ \leq\ pq/2^n$.* ∎

5 The SIV Mode of Operation

SIV MODE. Fix an n-bit blockcipher E and let $\Pi = \mathrm{CTR}$ be counter mode [27] over E, with an incrementing function of $S \mapsto \mathbf{2}S$ (that is, multiply by $\mathtt{x}$ in the finite field). Let $F = \mathrm{CMAC}^* = \mathrm{S2V}[\mathrm{CMAC}]$ be the result of applying the S2V construction to the CMAC [28], again with an underlying blockcipher of E. (Recall that CMAC is a NIST-recommended CBC MAC variant. It has a message space $\{0,1\}^*$.) Consider the scheme SIV$[F, \Pi]$. By combining Theorems 1 and 2 and known results about CMAC and CTR mode [3, 18], the suggested mechanism is a provably secure DAE assuming E is a secure PRP. The proven security falls off, as usual, in $\sigma^2/2^n$ where σ is the total number of blocks asked about. We overload the name SIV and call the mode of operation just described SIV mode. We emphasize that the only thing left unspecified in the definition of SIV mode is the underlying blockcipher, which would typically be AES.

COMMENTS. Comparing SIV-AES and the X9.102 scheme AESKW, say, we note that, with SIV-AES, (1) the message space and header space are now $\{0,1\}^*$ instead of unusual sets; (2) message expansion is now independent of header length and message length; (3) the number of blockcipher calls is reduced by a factor of at least six; (4) vector-valued headers can now be handled, and the contribution of any component can be pre-processed if it is to be held fixed; (5) one now has a provable-security guarantee, falling off in $\sigma^2/2^n$, where σ is the total number of message blocks acted on. On the other hand, there is an effective attack on SIV if one can ask this many message blocks, while we do not know if this is true for AESKW.

In the instantiation of SIV we could have used, in place of CMAC, the composition of a universal hash function that gives n-bit outputs with an n-bit blockcipher. This demonstrates that the DAE goal can be achieved by a single "cryptographic" pass over the plaintext, plus a universal-hash-function computation over the header and plaintext. Similarly, a parallelizable MAC like PMAC [10] could have been used in place of CMAC, illustrating that DAE can be achieved by a parallelizable scheme.

6 Misuse-Resistant AE

This section gives an application of DAEs motivated not by the key-wrap problem but by the goal of constructing symmetric encryption schemes that are resistant to misuse. We are specifically concerned with IV-misuse, meaning that the IV is used in a way other than the way mandated by the scheme; for example, using a counter when the scheme requires a random value, or repeating an IV when the scheme requires it to be a nonce. Experience has shown that IVs are frequently mishandled. An encryption scheme robust against misuse should at least be an AE scheme (as programmers, protocol designers, and even books often assume that encryption provides for authenticity) and so we will treat IV-misuse within the context of authenticated encryption and not privacy-only encryption. The notion is applicable to the latter context, too.

Designing an IV-based AE scheme that is secure when its IV is an arbitrary nonce—not just when it is a random value—is a first move in the direction of making schemes robust against IV-misuse. The current section takes this a step further; we aim for an AE scheme in which if the IV *is* a nonce then one achieves the usual notion for nonce-based AE; and if the IV *does* get repeated then authenticity remains and privacy is compromised only to the extent that some minimal amount of information may be revealed, the information being if this plaintext is equal to a prior one, and even that is revealed only if both the message and its header have been used with this particular IV. Our formalization will capture this intent.

Revised syntax for an IV-based encryption scheme. Let us update the syntax of a conventional IV-based encryption scheme to accommodate an associated header. In this case an IV-based encryption scheme is a tuple $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ where everything is as before except that the encryption algorithm and decryption algorithm take an extra argument: now they are deterministic algorithms that map $\mathcal{K} \times \{0,1\}^{**} \times \{0,1\}^* \times \{0,1\}^*$ to $\{0,1\}^* \cup \{\perp\}$. We write $\mathcal{E}_K(H, IV, X)$ or $\mathcal{E}_K^{H,IV}(X)$ in place of $\mathcal{E}(K, H, IV, X)$ and $\mathcal{D}_K(H, IV, C)$ or $\mathcal{D}_K^{H,IV}(Y)$ in place of $\mathcal{D}(K, H, IV, Y)$. There must be sets $\mathcal{H}$, $\mathcal{IV}$, and $\mathcal{X}$ such that $\mathcal{E}_K^{H,IV}(X) \in \{0,1\}^*$ iff $H \in \mathcal{H}$ and $IV \in \mathcal{IV}$ and $X \in \mathcal{X}$. We call $\mathcal{IV}$ the *IV space* of Π. We require that $\mathcal{D}_K^{H,IV}(Y) = X$ if $\mathcal{E}_K^{H,IV}(X) = Y$ and $\mathcal{D}_K^{H,IV}(Y) = \perp$ if there is no such X.

Misuse-resistant AE security. To measure the AE-security of an encryption scheme $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ in the face of possible IV-reuse, imagine an adversary that may ask any sequence of encryption queries, even those that repeat IVs, and any sequence of decryption queries, which may likewise repeat IVs. We want the encryption oracle to return bits that look random except when this is impossible—on a repeated triple of (header, IV, message)—and the decryption oracle should return $\perp$ except when the triple is already known to have a valid decryption. For simplicity, assume as before that our IV-based encryption scheme is length-preserving.

Definition 2. *Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be an IV-based encryption scheme that can handle an associated header and let A be an adversary. Then the* ***MRAE-advantage*** *of A in attacking Π is*

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

The adversary may not repeat a left-query and may not ask a right-query (H, IV, Y) *if some previous left-query* (H, IV, X) *returned* Y. ▮

Of course the $\mathcal{E}_K$ oracle returns $\mathcal{E}_K(H, IV, X)$ on input (H, IV, X) and $\mathcal{D}_K$ returns $\mathcal{D}_K(H, IV, Y)$ on input (H, IV, Y). As before $\$(H, IV, X)$ returns a random string of length $n + |X|$ and $\perp(\cdot, \cdot, \cdot)$ always returns $\perp$.

The MRAE-notion of security trivially implies nonce-based AE-scheme security: the latter is the special case where the adversary is not allowed to repeat an *IV* to any left query. Note that all proposed AE schemes to date [19, 21, 26, 29, 33] do fail should an IV get repeated: existing AE schemes are not MRAE-secure.

Building a misuse-resistant AE scheme. We can turn a DAE scheme $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ with header space $\{0,1\}^{**}$ and message space $\mathcal{X}$ into a misuse-resistant AE scheme $\widetilde{\Pi} = (\mathcal{K}, \widetilde{\mathcal{E}}, \widetilde{\mathcal{D}})$ by regarding the IV as one of the components, say the last component, of the header. In particular, SIV mode can be regarded as an MRAE scheme by asserting that one of the header components, say the last one specified, is an IV.

Correctness. Correctness of the MRAE scheme described above is nearly immediate. Given an adversary A for breaking the misuse-resistant AE scheme (it distinguishes $\mathcal{E}_K(\cdot, \cdot, \cdot),\ \mathcal{D}_K(\cdot, \cdot, \cdot)$ from $\$(\cdot, \cdot, \cdot),\ \perp(\cdot, \cdot, \cdot)$) we get a comparably good adversary B for breaking the DAE, distinguishing $\mathcal{E}_K(\cdot, \cdot),\ \mathcal{D}_K(\cdot, \cdot)$ from $\$(\cdot, \cdot),\ \perp(\cdot, \cdot)$: adversary B runs A and maps left queries (H, IV, X) to queries $(\langle H, IV \rangle, X)$, and maps right queries (H, IV, Y) to queries $(\langle H, IV \rangle, Y)$. The syntax and DAE-security notion for a PRI have been designed to "match up" so that there is nothing to do.

Comments. Since all we have done in the construction is to hijack a component of the header as an IV, it seems as though nothing has actually been done. Yet the MRAE goal is conceptually different from the DAE goal, the former employing an IV and gaining for this a stronger notion of security. The header and the IV are conceptually different, the one being user-supplied data that the user wants authenticated, the other being a mechanism-supplied value needed to obtain a strong notion of security.

In retrospect, it is easy to construct an MRAE scheme by a sequence of simple steps. One can achieve this goal in a trivial way from a DAE scheme that takes a vector-valued header. Such a DAE scheme is easily built from a vector-input PRF and an IND\$-secure conventional encryption scheme. At least if one is unconcerned with optimizing efficiency, a vector-input PRF is easily made from a string-input PRF. String-input PRFs and IND\$-secure conventional encryption schemes can be built from blockciphers by well-known means. So each step along our path is easy or well-known. Still, the direct construction of an MRAE or DAE scheme from a blockcipher is not a simple matter, as evidenced by the long history of buggy or baroque AE schemes Perhaps simple is how things seem *after* finding the right abstraction boundaries.

7 Properties of DAEs

This section investigates the properties of the DAE notion, looking in three different directions. First we explain the sense in which DAE achieve semantic security, indeed AE, when plaintexts carry a key. Next we give an alternative definition for DAEs, called pseudorandom injections, based on quite different intuition. Finally we show that the "all-in-one" definition of DAE can equivalently be factored into separate privacy and authenticity notions, as is traditionally done in this domain.

DAEs achieve semantic security when plaintexts carry a key. A folklore justification for using a key-wrap scheme instead of a conventional AE scheme is that, in the key-wrap setting, one expects the plaintext to carry a random cryptographic key, and so a probabilistic or stateful mechanism should not be needed. We sketch a result that validates this intuition.

A *key-insertion scheme* is a pair of algorithms $\Phi = (\text{InsertKey}, \text{ExtractKey})$, the first for inserting a κ-bit random value into a plaintext and the second for extracting it. Algorithm InsertKey, on input of $X \in \{0,1\}^*$, chooses a random $R \xleftarrow{\$} \{0,1\}^\kappa$ and returns $M \xleftarrow{\$} \text{InsertKey}(X)$. An equivalent viewpoint is that InsertKey is deterministic and takes the random string R as input; then we write $M \leftarrow \text{InsertKey}(X, R)$. Algorithm ExtractKey takes $M \in \{0,1\}^*$ and returns $\langle X, R\rangle$ with $|R| = \kappa$. Given a DAE $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ define the probabilistic encryption scheme $\widetilde{\Pi} = (\mathcal{K}, \widetilde{\mathcal{E}}, \widetilde{\mathcal{D}})$ by:

Algorithm $\widetilde{\mathcal{E}}_K(H, X)$	Algorithm $\widetilde{\mathcal{D}}_K(H, Y)$
$R \xleftarrow{\$} \{0,1\}^\kappa$	$M \leftarrow \mathcal{D}_K(H, Y)$
$M \leftarrow \text{InsertKey}(X, R)$	**if** $M = \bot$ **then return** $\bot$
if $M = \bot$ **return** $\bot$	**return** ExtractKey(M)
return $\mathcal{E}_K(H, M)$	

The encryption scheme is nonstandard insofar as decryption of a ciphertext Y returns not only the underlying plaintext X but also the random bits R that were inserted. Correspondingly, we must adapt the definition of AE to get a variant of $\mathbf{Adv}^{\text{ae}}_{\Pi}(A)$, call it $\mathbf{Adv}^{\text{kiae}}_{\Pi}(A)$, where when the adversary asks for the encryption of X we choose the random R and return it along with the ciphertext. This must look like random bits.

As long as the inserted key is sufficiently long, the algorithm described achieves the security notion we have sketched. We omit further details.

DAEs are equivalent to PRIs. A secure pseudorandom injection (PRI) resembles a random injective function with the desired amount of length-expansion. We allow a chosen-ciphertext attack in our definition (that is, we focus on a "strong" PRI, analogous to a strong PRP [24]), giving the adversary both the forward and backward direction of the function. We allow the PRI to be tweakable [23], so that the scheme can be used to authenticate an associated header. We allow the domain to be fairly arbitrary—in particular, we consider message spaces that contain strings of various lengths.

Formally, let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be a DAE with header space $\mathcal{H}$ and message space $\mathcal{X}$. Imagine an adversary A given access to two oracles—one for $\mathcal{E}$ and one for $\mathcal{D}$. We want to say that this pair looks just like a random injection f and its inverse f^{-1}, the random injection f having the same signature as $\mathcal{E}$. For $e\colon \mathcal{H} \times \mathcal{X} \to \mathbb{N}$ let $\mathsf{Inj}_e^{\mathcal{H}}(\mathcal{X}, \mathcal{Y})$ be the set of all injective functions f from $\mathcal{H} \times \mathcal{X}$ to $\mathcal{Y}$ such that $|f(H, X)| = |X| + e(H, X)$.

Definition 3. *Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be a DAE with header space $\mathcal{H}$, message space $\mathcal{X}$, and expansion e. The **PRI-advantage** of adversary A in breaking Π is*
$$\mathbf{Adv}_{\Pi}^{\mathrm{pri}}(A) = \Pr\Big[K \xleftarrow{\$} \mathcal{K} :\ A^{\mathcal{E}_K(\cdot,\cdot),\, \mathcal{D}_K(\cdot,\cdot)} \Rightarrow 1\Big] - \Pr\Big[f \xleftarrow{\$} \mathsf{Inj}_e^{\mathcal{H}}(\mathcal{X}, \mathcal{Y}) :\ A^{f(\cdot,\cdot),\, f^{-1}(\cdot,\cdot)} \Rightarrow 1\Big].$$
∎

The f^{-1} oracle above, on input (H, Y) returns the point X such that $f(H, X) = Y$; if there is no such point then it returns the distinguished value $\perp$. As before, we may assume without loss of generality that the adversary does not repeat a query, that it does not ask (H, Y) of its right oracle if some previous left oracle query (H, X) returned Y, that it does not ask (H, X) of its left oracle if some previous right-oracle query (H, Y) returned X, and that it does not ask any query (H, X) outside of $\mathcal{H} \times \mathcal{X}$.

Assuming a reasonable amount of stretch, the PRI and DAE notions of security are very close, as the following theorem shows.

Theorem 3. *Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be a DAE with header space $\mathcal{H}$, message space $\mathcal{X}$, and stretch s, and let $\tau = \min_{X \in \mathcal{X}}\{|X|\}$ be the length of a shortest plaintext. Let A be an adversary that asks at most q_{L} left-oracle queries, q_{R} right-oracle queries, for a total of $q = q_{\mathrm{L}} + q_{\mathrm{R}}$ queries. Then $\left|\mathbf{Adv}_{\Pi}^{\mathrm{pri}}(A) - \mathbf{Adv}_{\Pi}^{\mathrm{dae}}(A)\right| \le q^2/2^{s+\tau+1} + 4q_{\mathrm{R}}/2^s$.* ∎

In other words, as the stretch s grows, the DAE and PRI notions converge. The quantitative difference between the measures is small if the stretch is, say, $s = 128$ bits. Among other reasons, it is to achieve this equivalence with PRIs that our definition for them used indistinguishability from random bits rather than, say, indistinguishability from the encryption of random bits.

Proof. Let A be an adversary that has access to two oracles. Let it ask q_{L} queries of its left oracle and q_{R} queries of its right oracle, and let $q = q_{\mathrm{L}} + q_{\mathrm{R}}$. With the obvious notational simplifications we have

$$\begin{aligned}\left|\mathbf{Adv}_{\Pi}^{\mathrm{pri}}(A) - \mathbf{Adv}_{\Pi}^{\mathrm{dae}}(A)\right| &= \left|\Pr\left[A^{f(\cdot,\cdot),\, f^{-1}(\cdot,\cdot)} \Rightarrow 1\right] - \Pr\left[A^{\$(\cdot,\cdot),\, \perp(\cdot,\cdot)} \Rightarrow 1\right]\right| \\ &= \left|\Pr\left[A^{\mathrm{G1}} \Rightarrow 1\right] - \Pr[A^{\mathrm{G0}} \Rightarrow 1]\right|\end{aligned}$$

for the games G0 and G1 defined in Fig. 3. Recall that booleans are initialized to `false`, sets are initialized to empty, and partial functions are initialized to everywhere undefined with the symbol `undef`. The set $\mathrm{Image}(f(H, \cdot))$ contains all points $Y \neq$ `undef` such that $f(H, X) = Y$ for some $X \in \mathcal{X}$. Set difference is indicated with a minus sign. Look first at game G0. Much of the code (lines 12–13

On query left(H, X)**:**
10 $c \leftarrow |X| + e(H, X)$
11 $Y \xleftarrow{\$} \{0,1\}^c$
12 **if** $Y \in \text{Image}(f(H, \cdot)) \cup \mathit{Invalid}^H$ **then**
13 $\quad \mathit{bad} \leftarrow$ `true` , $Y \xleftarrow{\$} \{0,1\}^c - \text{Image}(f(H, \cdot)) - \mathit{Invalid}^H$
14 **return** $f(H, X) \leftarrow Y$

On query right(H, Y)**:**
20 $c \leftarrow |Y|$
21 $\mathit{EligibleX} \leftarrow \{X \in \{0,1\}^{\le c}\colon |X| + e(H, X) = |Y| \text{ and } f(H, X) = \texttt{undef}\}$
22 $\mathit{EligibleY} \leftarrow \{0,1\}^c - \text{Image}(f(H, \cdot)) - \mathit{Invalid}^H$
23 $x \xleftarrow{\$} [1 \,..\, |\mathit{EligibleY}|]$
24 **if** $x \in [1..|\mathit{EligibleX}|]$ **then**
25 $\quad \mathit{bad} \leftarrow$ `true` , $X \leftarrow$ the x^{th} string of $\mathit{EligibleX}$, $f(H, X) \leftarrow Y$, **return** X
26 $\mathit{Invalid}^H \leftarrow \mathit{Invalid}^H \cup \{Y\}$
27 **return** $\perp$

Fig. 3. Games used in the proof of Theorem 3. Game G1 is the complete code; game G0 omits the shaded statements.

and 20–26) is irrelevant to what the adversary sees. Each query left(H, X) returns a random string of $|X| + e(H, X)$ bits and each query right(H, Y) returns $\perp$. Thus game G0's (left, right) oracles faithfully simulate a pair of oracles $(\$, \perp)$ and we have that $\Pr[A^{\text{G0}} \Rightarrow 1] = \Pr[A^{\$,\perp} \Rightarrow 1]$.

Game G1 is more subtle. We claim that its (left, right) oracles are simply a lazy evaluation of a pair of oracles (f, f^{-1}) with the desired domain and range. To see this, understand first that the partial function $f(H, \cdot)$ maintains the correspondence $X \mapsto f(H, X)$ for those domain points that we have already assigned values to, while the set $\mathit{Invalid}^H$ maintains the set of points Y that have become ineligible to be $f(H, X)$ values, for any X, by virtue of having been asked right(H, Y) and having returned $\perp$, effectively asserting that $f^{-1}(H, Y) = \perp$ and so Y is outside the image of $f(H, \cdot)$. Now, starting at left(H, X) queries, we begin at line 10 by calculating the length c of the ciphertext that we must return. The code at lines 11–14 returns a random string Y of length c subject to the constraint that Y is outside of the image of $f(H, \cdot)$ and not ineligible to be an $f(H, X)$ value by virtue of having asserted that there is no preimage for Y with tweak H. Looking next at right(H, Y) queries, we calculate at line 21 the set *EligibleX* of values X that could possibly map to Y using tweak H, and we calculate at line 22 the set of strings Y that could, at this moment be paired with strings in *EligibleX*. By our conventions on the adversary making no "pointless" queries, the string Y will necessarily be among the strings in $\mathit{EligibleY}$. Since we aim to randomly and injectively pair points in *EligibleX* with points in $\mathit{EligibleY}$, the chance that a given point Y in $\mathit{EligibleY}$ has a preimage in *EligibleX* is just $|\mathit{EligibleX}|/|\mathit{EligibleY}|$. Lines 23 and 24 effectively flip a coin with this bias, deciding if the string $Y \in \mathit{EligibleY}$ should or should not be given a (random) preimage in *EligibleX*. If it is not given a preimage, we record this decision by augmenting $Invalid^H$ at line 26. If it is given a preimage, it is given a random one by lines 23–25, the choice is recorded, and the random preimage is returned. We have thus provided a perfect simulation of an (f, f^{-1}) oracle, and so $\Pr[A^{G1} \Rightarrow 1] = \Pr[A^{f,f^{-1}} \Rightarrow 1]$.

To bound $|\Pr[A^{G1} \Rightarrow 1] - \Pr[A^{G0} \Rightarrow 1]|$ we can now invoke the fundamental lemma of game-playing [7], since games G1 and G0 have been defined to be identical apart from the sequel of statements $bad \leftarrow \texttt{true}$. The lemma assures us that $|\Pr[A^{G1} \Rightarrow 1] - \Pr[A^{G0} \Rightarrow 1]| \leq \Pr[A^{G0} \text{ sets } bad\,]$.

Let BAD be the event that A^{G0} causes *bad* to get set to `true`. We must bound the probability of BAD. Remember that the shaded statements have been expunged from the game. Prior to BAD occurring, each left-query adds a single point to a set $\mathrm{Image}(f(H,\cdot))$ but has no impact on any set $Invalid^H$, while each right-query adds a single point to a set $Invalid^H$ but has no impact on any set $\mathrm{Image}(f(H,\cdot))$. If the i^{th} query is left-query then the set $\mathrm{Image}(f(H,\cdot)) \cup Invalid^H$ will have at most $i-1$ points and the chance that *bad* will get set at line 13 will be at most $(i-1)/2^{s+\tau}$ and so, overall, the probability that *bad* gets set at line 13 is at most $\sum_{i=1}^{q}(i-1)/2^{s+\tau} \leq q^2/2^{s+\tau+1}$. If the i^{th} query is a right-query then *bad* will be set with probability $|EligibleX|/|EligibleY|$ for the current sets *EligibleX* and *EligibleY*. How big can $|EligibleX|$ be? Asked a query Y of length c, even if *every* string of length at most $c-s$ (the maximal possible length) is in *EligibleX*, still we will have that $|EligibleX| < 2^{c+1-s}$. Conversely, how small can $|EligibleY|$ be? On the i^{th} query we know that $|EligibleY| > 2^c - i$. So on the i^{th} query we have that $|EligibleX|/|EligibleY| < 2^{c+1-s}/(2^c - i) \leq 2^{2-s}$ assuming $i \leq 2^{c-1}$ or, more strongly, assuming $q \leq 2^{s+\tau-1}$. Summing over all q_{R} right-queries we have that the probability that *bad* gets set at line 25 is at most $4q_{\mathrm{R}}/2^s$. Since the result becomes vacuous when $q > 2^{s+\tau-1}$, we may now drop that technical condition and conclude the theorem.

Equivalence of all-in-one and two-requirement definitions. To define DAE-security one could specify separate notions for deterministic privacy, detPriv, and deterministic authenticity, detAuth, and demand both. This "dual-requirement" approach is the one that has been taken in all prior work on AE. In our setting one could let $\mathbf{Adv}_{\Pi}^{\mathrm{detPriv}}(A) = \Pr[A^{\mathcal{E}_K(\cdot,\cdot)} \Rightarrow 1] - \Pr[A^{\$(\cdot,\cdot)} \Rightarrow 1]$ and $\mathbf{Adv}_{\Pi}^{\mathrm{detAuth}}(A) = \Pr[A^{\mathcal{E}_K(\cdot,\cdot),\, \mathcal{D}_K(\cdot,\cdot)} \text{ forges}]$ where in the first definition A does not repeat a query, and in the second it never asks a right-query (H,Y) having already asked a left-query (H,X) that returned Y. Saying that A forges means that it asks a right-query (H,Y) and gets a response other than $\perp$, and A did not earlier ask a left-query (H,X) that returned Y.

It is straightforward to prove that our all-in-one notion of DAE-security and the two-requirement definition just sketched are equivalent. We omit further details.

The idea above can be extended to other variants of AE: the encryption scheme may be probabilistic, nonce-based, or deterministic; the privacy requirement can be indistinguishability from random bits or conventional indistinguishability; and message headers may be present or absent, strings or vectors.

For any of these variants one can give a two-requirement definition or an all-in-one definition. In all cases we have investigated, the results come out as above: the all-in-one definition and the two-requirement definition are equivalent.

All-in-one definitions for AE resemble the definition for chosen-ciphertext-attack (CCA2) security [3, 4]; the definition of AE strengthens CCA2 in a simple and natural way. Perhaps it is only historical accident that our community has come to think of AE as privacy+authenticity and not as "CCA3 security."

Acknowledgments

Many thanks to the X9F1 working group, whose draft standard motivated this paper, and Morris Dworkin, who made this work known to us [13]. Thanks to Jesse Walker for an enormous number of valuable comments; to Susan Langford for noticing a significant error in an earlier draft; to Steve Bellovin for voicing his concerns about IV-misuse at a meeting back in 2000 (his comments ultimately motivated Section 6); Mihir Bellare for his typically perceptive comments; and the Eurocrypt 2006 PC for their comments. Phil Rogaway was supported by NSF 0208842 and a gift from Intel Corp. Much of this paper was written while Rogaway was a visitor to the School of Information Technology at Mae Fah Luang University, Thailand. Many thanks to MFLU and, in particular, to Dr. Thongchai Yooyativong and Dr. Tatsanee Mallanoo, for their generous hospitality.

References

1. J. An and M. Bellare. Does encryption with redundancy provide authenticity? *Advances in Cryptology – Eurocrypt '01*, LNCS vol. 2045, Springer, pp. 512–528, 2001.
2. M. Bellare, A. Boldyreva, L. Knudsen, and C. Namprempre. On-Line ciphers and the Hash-CBC constructions. *Advances in Cryptology – Crypto '01*, LNCS vol. 2139, Springer, pp. 292–309, 2001.
3. M. Bellare, A. Desai, E. Jokipii, and P. Rogaway. A concrete security treatment of symmetric encryption: analysis of the DES modes of operation. *Proc. of the 38th Symposium on Foundations of Computer Science*, IEEE Press, pp. 394–403, 1997.
4. M. Bellare, A. Desai, D. Pointcheval, and P. Rogaway. Relations among notions of security for public-key encryption schemes. *Advances in Cryptology – Crypto'98*, LNCS vol.1462, Springer, pp. 26–45, 1998.
5. M. Bellare, J. Kilian, and P. Rogaway. The security of the cipher block chaining message authentication code. *J. of Computer and System Science (JCSS)*, vol. 61, no. 3, pp. 362–399, Dec 2000.
6. M. Bellare and C. Namprempre. Authenticated encryption: relations among notions and analysis of the generic composition paradigm. *Advances in Cryptology – Asiacrypt '00*, LNCS vol. 1976, Springer, pp. 531–545, 2000.
7. M. Bellare and P. Rogaway. Code-based game-playing proofs and the security of triple encryption. Cryptology ePrint report 2004/331, 2004.

8. M. Bellare and P. Rogaway. Encode-then-encipher encryption: how to exploit nonces or redundancy in plaintexts for efficient encryption. *Advances in Cryptology – Asiacrypt '00*, LNCS vol. 1976, Springer, pp. 317–330, 2000.
9. M. Bellare, P. Rogaway, and D. Wagner. The EAX mode of operation. *Fast Software Encryption (FSE 2004)*, LNCS vol. 3017, Springer, pp. 389–407, 2004.
10. J. Black and P. Rogaway. A block-cipher mode of operation for parallelizable message authentication. *Advances in Cryptology – Eurocrypt '02*, LNCS vol. 2332, Springer, pp. 384-397, 2001.
11. J. Black and P. Rogaway. CBC MACs for arbitrary-length messages: the three-key constructions. *Advances in Cryptology – Crypto '00*, LNCS vol. 1880, Springer, pp. 197–215, 2000.
12. Y. Dodis and A. Smith. Entropic security and the encryption of high entropy messages. *Theory of Cryptography (TCC 2005)*, LNCS vol. 3378, Springer, pp. 556-577, 2005.
13. M. Dworkin. Request for review of key wrap algorithms. Cryptology ePrint report 2004/340, 2004. Contents are excerpts from a draft standard of the Accredited Standards Committee, X9, entitled *ANS X9.102 — Wrapping of Keys and Associated Data.*
14. O. Goldreich, S. Goldwasser, and S. Micali, How to construct random functions. *Journal of the ACM,* vol. 33, no. 4, pp. 210–217, 1986.
15. S. Goldwasser and S. Micali. Probabilistic encryption. *J. Comput. Syst. Sci.*, vol. 28, no. 2, pp. 270–299, 1984.
16. S. Halevi and P. Rogaway. A tweakable enciphering mode. *Advances in Cryptology – Crypto '03*, LNCS vol. 2727, Springer, pp. 482–499, 2003.
17. R. Housley. Triple-DES and RC2 key wrapping. IETF RFC 3217, Dec. 2001. Earlier version in RFC 2630, June 1999.
18. T. Iwata and K. Kurosawa. OMAC: One-key CBC MAC. *Fast Software Encryption (FSE 2003)*, LNCS vol. 2887, Springer, pp. 129–153, 2003.
19. C. Jutla. Encryption modes with almost free message integrity. *Advances in Cryptology – Eurocrypt '01*, LNCS vol. 2045, Springer, pp. 529–544, 2001.
20. J. Katz and M. Yung. Unforgeable encryption and adaptively secure modes of operation. *Fast Software Encryption (FSE 2000)*, LNCS vol. 1978, Springer, pp. 284–299, 2000.
21. T. Kohno, J. Viega, and D. Whiting. CWC: A high-performance conventional authenticated encryption mode. *Fast Software Encryption (FSE 2004)*, LNCS vol. 3017, Springer, pp. 427–445, 2004.
22. H. Krawczyk. The order of encryption and authentication for protecting communications (or: how secure is SSL?) *Advances in Cryptology – Crypto '01*, LNCS vol. 2139, Springer, pp. 310–331, 2001.
23. M. Liskov, R. Rivest, and D. Wagner. Tweakable block ciphers. *Advances in Cryptology – Crypto '02*, LNCS vol. 2442, Springer, pp. 31–46, 2002.
24. M. Luby and C. Rackoff. How to construct pseudorandom permutations from pseudorandom functions. *SIAM J. Comput.*, vol. 17, no. 2, pp. 373–386, 1988.
25. S. Matyas. Key handling with control vectors. *IBM Systems Journal*, vol. 30, no. 2, pp. 151–174, 1991.
26. D. McGrew and J. Viega. The Galois/Counter mode of operation (GCM). Manuscript, May 2005. Available from the NIST website.
27. National Institute of Standards and Technology, M. Dworkin, author. Recommendation for block cipher modes of operation, methods and techniques. NIST Special Publication 800-38A, 2001.

28. National Institute of Standards and Technology, M. Dworkin, author. Recommendation for block cipher modes of operation: the CMAC mode for authentication. NIST Special Publication 800-38B, May 2005.
29. National Institute of Standards and Technology, M. Dworkin, author. Recommendation for block cipher modes of operation: the CCM mode for authentication and confidentiality. NIST Special Publication 800-38C, May 2004.
30. D. Phan and D. Pointcheval. About the security of ciphers (semantic security and pseudo-random permutations). *Selected Areas in Cryptography (SAC 2004)*, LNCS vol 3357, Springer, pp. 182-197, 2004.
31. P. Rogaway. Authenticated-encryption with associated-data. *Proceedings of the 9th Annual Conference on Computer and Communications Security (CCS-9)*, ACM, pp. 98–107, 2002.
32. P. Rogaway. Nonce-based symmetric encryption. *Fast Software Encryption (FSE 2004)*, LNCS vol. 3017, Springer, pp. 348–359, 2004.
33. P. Rogaway, M. Bellare, and J. Black. OCB: A block-cipher mode of operation for efficient authenticated encryption. *ACM Transactions on Information and System Security* (TISSEC), vol. 6, no. 3, pp. 365–403, Aug. 2003.
34. A. Russell and H. Wong. How to fool an unbounded adversary with a short key. *Advances in Cryptology – Eurocrypt '02*, LNCS vol. 2332, Springer, pp. 133–148, 2002.
35. R. Schroeppel. The hasty pudding cipher. AES candidate submitted to NIST, 1998.
36. S/MIME Working Group, IETF. Mailing list archives, 1997. http://www.imc.org/ietf-smime/index.html