Authenticated and Misuse-Resistant Encryption of Key-Dependent Data

Mihir Bellare and Sriram Keelveedhi

Department of Computer Science & Engineering, University of California San Diego,
9500 Gilman Drive, La Jolla, California 92093, USA.
http://www.cs.ucsd.edu/users/{mihir,skeelvee}/

Abstract. This paper provides a comprehensive treatment of the security of authenticated encryption (AE) in the presence of key-dependent data, considering the four variants of the goal arising from the choice of universal nonce or random nonce security and presence or absence of a header. We present attacks showing that universal-nonce security for key-dependent messages is impossible, as is security for key-dependent headers, not only ruling out security for three of the four variants but showing that currently standarized and used schemes (all these target universal nonce security in the presence of headers) fail to provide security for key-dependent data. To complete the picture we show that the final variant (random-nonce security in the presence of key-dependent messages but key-independent headers) is efficiently achievable. Rather than a single dedicated scheme, we present a RO-based transform RHtE that endows *any* AE scheme with this security, so that existing implementations may be easily upgraded to have the best possible seurity in the presence of key-dependent data. RHtE is cheap, software-friendly, and continues to provide security when the key is a password, a setting in which key-dependent data is particularly likely. We go on to give a key-dependent data treatment of the goal of misuse resistant AE. Implementations are provided and show that RHtE has small overhead.

1 Introduction

The key used by BitLocker to encrypt your disk may reside on the disk. The key under which a secure filesystem is encrypted may itself be stored in a file on the same system. The result is encryption of key-dependent data.

There is growing recognition that security of key-dependent data, first defined to connect cryptography to formal methods [18] and provide anonymous credentials [24], is a more direct and widespread concern for secure systems. The problem is particularly acute when keys are passwords, for many of us store our passwords on our systems and systems store password hashes. If nothing else, one cannot expect applications to ensure or certify that their data is *not* key-dependent, making security for key-dependent data essential for easy-to-use, robust and misuse-resistant cryptography.

This paper provides a comprehensive treatment of security for key-dependent data for the central practical goal of symmetric cryptography, namely authenticated encryption. For each important variant of the goal we either show that it is impossible to achieve security or present an efficient solution. Our attacks rule out security for in-use and standardized schemes in their prescribed and common modes while our solutions show how to adapt them in minimal ways to achieve the best achievable security. Let us now look at all this more closely.

Background. The standard IND-CPA and IND-CCA goals that our encryption schemes are proven to meet do not guarantee security when the message being encrypted depends on the key. (In the symmetric setting, we mean the single key used for both encryption and decryption.) Black, Rogaway and Shrimpton (BRS) [18] extend IND-CPA to allow key-dependent messages (KDMs). The adversary provides its encryption oracle with a function ϕ, called a message-deriving function, that the game applies to the target key K to get a message M, and the adversary is returned either an encryption of M under K or the encryption of $0^{|M|}$, and must be unable to tell which. (They, and we, actually consider a multi-key setting, but the single-key setting will simplify the current discussion.) They present a simple random-oracle (RO) model solution.

Post-BRS work has aimed mainly at showing existence of schemes secure against as large as possible a class of message deriving functions without random oracles [19, 36, 4, 21, 23, 20, 7, 25, 17, 3, 38]. The schemes suffer from one or more of the following: they are in the asymmetric setting while data encryption in practice is largely symmetric; they are too complex to consider usage; or security is provided for a limited, mathematical class of message-deriving functions which does not cover all key-dependencies in systems.

Backes, Pfitzmann and Scedrov (BPS) [6] define KDM-security for a basic form of authenticated encryption and and show that Encrypt-then-MAC [12] achieves it if the encryption scheme is KDM secure and the MAC is strongly unforgeable (remarkably, no KDM security is required from the MAC), resulting in RO model solutions via [18]. In this paper we will extend their treatment of AE in several directions.

Setting. Privacy without authenticity, meaning plain (IND-CPA) encryption, is of limited utility. The most important symmetric primitive in practice is authenticated encryption (AE), which provides both privacy and integrity. This is evidenced by numerous standards and high usage: CCM [50, 49] is in IEEE 802.11, IEEE 802.15.4, IPSEC ESP and IKEv2; GCM [39] is standardized by NIST as SP 800-38D; EAX [16] is in ANSI C12.22 and ISO/IEC 19772; OCB 2.0 [45, 47] is in ISO 19772. Consideration of KDM security for these standards is compelling and urgent but has not been done. We seek to fill this gap.

Symmetric encryption schemes take as input a nonce, also called an IV. Classically — [9] following [30]— this was chosen at random by the encrypter. We call this random-nonce security (`r`). Later schemes targeted universal-nonce security (`u`) [44, 46, 48] where security must hold even when the adversary provides the nonce, as long as no nonce is re-used. This is adopted by the above-mentioned standards.

	(ki, ki)	(kd, kd)	(ki, kd)	(kd, ki)
u	Yes	No	No	No
r	Yes	No	No	Yes

Fig. 1. Each of message and header may be key-dependent (kd) or key-independent (ki), leading to the four choices naming the columns. Security could be universal-nonce (u) or random-nonce (r), leading to the two choices naming the rows. For each of the 8 possibilities, we indicate whether security is possible (Yes, meaning a secure scheme exists) or impossible (No, meaning there is an attack that breaks *any* scheme in this category). The first column reflects known results when inputs are not key-dependent.

Besides key, nonce and message, modern AE schemes, including the above standards, take input a header, or associated data [44]. The scheme must provide integrity but *not* privacy of the header. Thus we must consider that not just the message, but also the header, could be key-dependent.

Abbreviate key-dependent by kd and key-independent by ki. With two choices for nonce type —$\mathtt{nt} \in \{\mathtt{u}, \mathtt{r}\}$— two for message type —$\mathtt{mt} \in \{\mathtt{kd}, \mathtt{ki}\}$— and two for header type —$\mathtt{ht} \in \{\mathtt{kd}, \mathtt{ki}\}$— we have 8 variants of AE. The form of AE treated by Backes, Pfitzmann and Scedrov [6] is the special case of $(\mathtt{nt}, \mathtt{mt}, \mathtt{ht}) = (\mathtt{r}, \mathtt{kd}, \mathtt{ki})$ in which the header is absent.

Definition. Our first contribution is a definition of security for AE under key-dependent inputs that captures all these 8 variants in a unified way. The encryption oracle takes functions ϕ_m, ϕ_h, and applies them to the key to get message and header respectively, and the adversary gets back either an encryption of these under the game-chosen target key, or a random string of the same length. The decryption oracle takes a ciphertext and, importantly, not a header but a function ϕ_h to derive it from the key, and either says whether or not decryption under the key is valid, or always says it is invalid. Varying the way nonces are treated and from what spaces ϕ_m, ϕ_h are drawn yields the different variants of the notion. A definition of MACs for key-dependent messages emerges as the special case of empty messages.

On a real system, the data may be a complex function of the key, such as a compressed (zipped) version of file containing, amongst other things, the key, or an error-corrected version of the key. If the key is a password the system will store its hash that will be encrypted as part of the disk, so common password-hashing functions must be included as message-deriving functions. All this argues for not restricting the types of message-deriving or header-deriving functions, and indeed, following [18, 6], we allow any functions in this role. These functions are even allowed to call the RO, a source of challenges in proofs.

Underlying the above definition is a new one of the standard AE goal that simplifies that of [48] by having the decryption oracle turn into a verification oracle, returning, not the full decryption, but only whether it succeeded or not, along the lines of [12]. When data is key-independent, these and prior formulations [12, 37] are equivalent, but the difference is important with key-dependent data.

Impossibility results. We present an attack that shows that *no* AE scheme can achieve universal-nonce security for key-dependent data. (Regardless of whether or not the header is key-dependent.) This explains the "No" entries in the first row of Fig. 1. The attack requires only that the nonce is predictable. Thus it applies even when the nonce is a counter, ruling out KDM security for counter-based AE schemes and showing that the standardized schemes (CCM, GCM, EAX, OCB) are all insecure for key-dependent messages in this case. The attack does not use the decryption oracle, so rules out even KDM universal-nonce CPA secure encryption. Thus, the universal-nonce security proven for the standardized schemes for key-independent messages fails to extend to key-dependent ones, demonstrating that security for key-dependent messages is a fundamentally different and stronger security requirement.

An attack aiming to show that no stateful scheme is KDM-CPA secure was described in [18] but the message-deriving functions execute a search and it is not clear how long this will take to terminate or whether it will even succeed. (In asymptotic terms, the attack is not proven to terminate in polynomial time.) Our attack extends theirs to use pairwise independent hash functions, based on which we prove that it achieves a constant advantage in a bounded (polynomial) amount of time. Interestingly, as a corollary of the bound proven on our *modified* attack, we are able to also prove a bound on the running time of the attack of [18], although it was not clear to us how to do this directly.

We also present an attack that shows that *no* AE scheme can achieve security for key-dependent headers. (Even for random, rather than universal, nonce security, and even for key-independent messages.) This explains the "No" entries in columns 2 and 3 of Fig. 1. This rules out security of the standardized schemes even with random nonces in a setting where headers may be key-dependent.

One might consider this trivial with the following reasoning: "Since the header is not kept private, the adversary sees it, and if it is key-dependent, it could for example just be the key, effectively giving the adversary the key." The fallacy is the assumption that the adversary sees the header. In our model, it is given a ciphertext but not directly given the header on which the ciphertext depends. This choice of model is not arbitrary but reflects applications, where a key-dependent header is present on the encrypting and decrypting systems (which may be the same system) but not visible to the adversary. Instead, the attack exploits the ability of the adversary to test validity of ciphertexts with implicitly specified headers.

RHtE. We turn to achieving security in the only viable, but still important setting, namely $(\mathtt{nt}, \mathtt{mt}, \mathtt{ht}) = (\mathtt{r}, \mathtt{kd}, \mathtt{ki})$. As background, recall that to achieve KDM-CPA security, BRS [18] encrypt message M by picking R at random and returning $H(K\|R)\oplus M)$ where H is a RO returning $|M|$ bits. (Here and below, it is assumed the decryptor and adversary also get the nonce R, but it is not formally part of the ciphertext.) We note that this is easily extended to achieve $(\mathtt{r}, \mathtt{kd}, \mathtt{ki})$-AE security. To encrypt header H and message M under key K, pick R at random and return (C, T) where $C = H_1(K\|R)\oplus M$ and $T = H_2(K\|R, H, C)$ and H_1, H_2 are ROs.

Randomized Hash then Encrypt (RHtE) is more practical. Unlike the above, it is not a dedicated scheme but rather transforms a standard (secure only for key-independent data) base AE scheme into a $(\mathtt{r}, \mathtt{kd}, \mathtt{ki})$-secure AE scheme. RHtE, given key L and randomness R, derives subkey $K = H(R\|L)$ via RO H and then runs the base scheme with key K on the header and message to get the ciphertext C. Only one-time security of the base scheme is required, so it could even be deterministic. The software changes are non-intrusive since the code of the base scheme is used unchanged. Thus RHtE can easily be put on top of existing standards like CCM, GCM, EAX, OCB to add security in the presence of key-dependent messages. As long as these base schemes transmit their nonce, RHtE has *zero overhead in bandwidth* because it can use the base scheme with some fixed, known nonce and use the nonce space for R. (It is okay to re-use the base-scheme nonce because this scheme is only required to be one-time secure. Its key is changing with every encryption.) The computational overhead of RHtE is independent of the lengths of header and message and hence becomes negligible as these get longer.

The proof of security is surprisingly involved due to a combination of three factors. First is that the message-deriving functions are allowed to call the RO. Second, while the BRS scheme and its extension noted above are purely information theoretic, the security of RHtE is computational due to the base scheme, and must be proven by reduction. Third, unlike BRS, we must deal with decryption queries. To handle all this we will need to invoke the security of the base scheme in multiple, inter-related ways, leading to a proof with two, interleaved hybrids that go in opposite directions.

Some indication of the complexity of the proof is provided by the fact that the bound we finally achieve in Theorem 5 is weaker than we would like. It is an interesting open problem to either prove a better bound for RHtE or provide an alternative scheme with such a bound.

Extensions. In filesystem encryption, as with most applications, security is likely to stem from your password pw. The system stores a hash $\overline{\mathsf{pw}} = h(\mathsf{pw})$ of it to authenticate you and an AE scheme must then encrypt or decrypt using pw. Key dependent data is now an even greater concern. One reason is that users tend to write their passwords in files in their filesystems. The other reasons is that $\overline{\mathsf{pw}}$ is a function of pw that must be stored on the system and thus will be encrypted with disk encryption. To address this, we show that RHtE is secure even when its starting key L is a password as long as the latter is drawn from a space that, asymptotically, has super-logarithmic min-entropy.

The security discussed so far relies crucially on using fresh randomness with each encryption. This is fine in theory but in real systems, failures of random-number generation (RNG) due to poorly gathered entropy or bugs are all too common and have lead to major security violations [29, 33, 22, 42, 41, 1, 51, 27]. Simply asking that system designers get their RNGs "right" is unrealistic. Misuse-resistant [18] or hedged [8] encryption take a different approach, mitigating the damage caused by RNG failures by providing as much security as possible when randomness fails.

We extend this to the key-dependent data setting in the full version [10]. A misuse resistant AE scheme for key-dependent data provides two things. First, it must continue to provide $(\mathtt{r}, \mathtt{kd}, \mathtt{kd})$-security when the nonce is random. Second, even for nonces that are entirely adversary controlled (and may repeat), the scheme must meet a second condition that we define to capture its providing the security of deterministic AE in the presence of key-dependent data. In the latter case it is impossible to protect against certain classes of message-deriving functions. We show however that RHtE provides security against any class of functions satisfying the output-unpredictability and collision-resistance conditions of [11]. This is a fairly significant class, containing functions of pragmatic interest.

Implementation. We implemented RHtE for base schemes CCM, EAX and GCM, with SHA256 instantiating the RO. The results, provided in [10]show for example that with CCM the slowdown is 11% for 5KB messages and only 1% for 50KB messages. The implementations use the crypto++ library on a Intel Core i5 M460 CPU running at 2.53 GHz with code compiled using g++ -O3 for data sizes small enough to fit in the level 2 cache.

Related work. The issue (key-dependent messages) was pointed out as early as Goldwasser and Micali [30], and asymmetric encryption of decryption keys was treated by Camenisch and Lysyanskaya [24], but a full treatment of key-dependent message (KDM) encryption awaited BRS [18], who provided RO model KDM-CPA secure schemes. Researchers then asked for what classes of message-deriving functions one could achieve KDM security in the standard model, providing results for both symmetric and asymmetric encryption under different assumptions [19, 36, 4, 21, 23, 20, 7, 25, 17, 3, 38]. On the more practical side, Backes, Dürmuth and Unruh [5] show that RSA-OAEP [13, 28] is KDM-secure in the RO model. Backes, Pfitzmann and Scedrov [6] treat active attacks and provide and relate a number of different notions of security.

By showing that IND-CPA security does not even imply security for the encryption of 2-cycles, Acar, Belenkiy, Bellare and Cash [2] and Green and Hohenberger [32] settled a basic question in this area and showed that achieving even weak KDM-security *requires* new schemes, validating previous efforts in that direction. Acar et. al. [2] also connect KDM secure encryption to cryptographic agility. Haitner and Holenstein [34] study the difficulty of proving KDM security by blackbox reduction to standard primitives.

Halevi and Krawczyk [35] consider blockciphers under key-dependent inputs. Muñiz and Steinwandt [40] study KDM secure signatures. González, in an unpublished thesis [31], studies KDM secure MACs.

Motivated by attacks on SSH, Paterson and Watson [43] consider notions of security (in the standard $\mathtt{ki}$-data context) which allow the attacker to interact in a byte-by-byte manner with the decryption oracle. Our treatment does not encompass such attacks, and extending the model of [43] to allow key-dependent data is an interesting direction for future work.

2 Definitions

We provide a unified definition for universal and random nonce AE security and then extend this to definitions of universal and random nonce AE security in the presence of key-dependent messages and headers.

NOTATION. If S is a (finite) set then $s \leftarrow\!\!\$\ S$ denotes the operation of picking s from S at random and $|S|$ is the size of S. Read the term "efficient" as meaning "polynomial-time" in the natural asymptotic extension of our concrete framework. If x is a string then $|x|$ denotes its length and $x[i]$ denotes its i-th bit. The empty string is denoted ε. By $a_1 \| \ldots \| a_n$, we denote the concatenation of strings $a_1, \ldots, a_n$. Unless otherwise indicated, an algorithm may be randomized. We denote by $y \leftarrow\!\!\$\ A(x_1, x_2, \ldots)$ the operation of running A on the indicated inputs and fresh random coins to get an output denoted y. For integers k, w let $\mathsf{Fun}(k, w)$ be the set of all functions ϕ for which there exists an integer $\mathsf{ol}(\phi)$, called the output length of ϕ, such that ϕ: $(\{0,1\}^k)^w \to \{0,1\}^{\mathsf{ol}(\phi)}$. Input-deriving functions will be drawn from this set. Let $\mathsf{Cns}(k, w)$ be the subset of $\mathsf{Fun}(k, w)$ consisting of constant functions, restricting attention to which drops KDI (key-dependent input) notions of security down to their standard, non-KDI counterparts.

GAMES. Some of our definitions and proofs are expressed via code-based games [15]. Such a game —see Fig. 2 for an example— consists of procedures that respond to adversary oracle queries. In an execution of game G with an adversary A, the latter must make exactly one INITIALIZE query, this being its first oracle query, and exactly one FINALIZE query, this being its last oracle query. In between, it can query other game procedures. Each time it makes a query, the corresponding game procedure executes, and what it returns, if anything, is the response to A's query. The output of FINALIZE, denoted G^A, is called the output of the game, and we let "$\mathrm{G}^A \Rightarrow d$" denote the event that this game output takes value d. If FINALIZE is absent it is understood to be the identity function, so the game output is the adversary output. Boolean flags are assumed initialized to false and $\mathsf{BAD}(\mathrm{G}^A)$ is the event that the execution of game G with adversary A sets flag bad to true. The running time of an adversary by convention is the worst case time for the execution of the adversary with the game defining its security, so that the time of the called game procedures is included.

AE SYNTAX. A symmetric encryption scheme $\mathsf{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ is specified by a key generation algorithm $\mathcal{K}$ that returns k-bit strings, an encryption function $\mathcal{E}$: $\{0,1\}^k \times \{0,1\}^n \times \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^*$ and decryption function $\mathcal{D}$: $\{0,1\}^k \times \{0,1\}^n \times \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^* \cup \{\perp\}$. Inputs to $\mathcal{E}$ are key, nonce, header and message, and output is a ciphertext. Inputs to $\mathcal{D}$ are key, nonce, header and ciphertext, and output is a message or $\perp$. We refer to k as the keylength and n as the noncelength. Both $\mathcal{E}$ and $\mathcal{D}$ are deterministic, it being the way nonces are handled by the games defining security that will distinguish universal-nonce and random-nonce security. We require that $\mathcal{D}(K, N, H, \mathcal{E}(K, N, H, M)) = M$ for all values of the inputs shown. We also require that $\mathcal{E}$ is length respecting in the sense that the length of a ciphertext

proc INITIALIZE // $\mathrm{KIAE}_{\mathsf{SE},\mathtt{nt}}$	proc INITIALIZE(w) // $\mathrm{KDAE}_{\mathsf{SE},\mathtt{nt}}$
$K \leftarrow\!\!\$\ \mathcal{K}$	For $j = 1, \ldots, w$ do
$S \leftarrow \emptyset$	$\quad K_j \leftarrow\!\!\$\ \mathcal{K}$; $S_j \leftarrow \emptyset$
$b \leftarrow\!\!\$\ \{0,1\}$	$b \leftarrow\!\!\$\ \{0,1\}$
proc ENC(N, H, M) // $\mathrm{KIAE}_{\mathsf{SE},\mathtt{nt}}$	proc ENC(j, N, ϕ_h, ϕ_m) // $\mathrm{KDAE}_{\mathsf{SE},\mathtt{nt}}$
If $(\mathtt{nt} = \mathtt{r})$ then $N \leftarrow\!\!\$\ \{0,1\}^n$	$M \leftarrow \phi_m(K_1, \ldots, K_w)$; $H \leftarrow \phi_h(K_1, \ldots, K_w)$
If $(b = 1)$ then $C \leftarrow \mathcal{E}(K, N, H, M)$	If $(\mathtt{nt} = \mathtt{r})$ then $N \leftarrow\!\!\$\ \{0,1\}^n$
Else $c \leftarrow \mathsf{cl}(\lvert M\rvert, \lvert H\rvert)$; $C \leftarrow\!\!\$\ \{0,1\}^c$	If $(b = 1)$ then $C \leftarrow \mathcal{E}(K_j, N, H, M)$
$S \leftarrow S \cup \{(N, H, C)\}$	Else $c \leftarrow \mathsf{cl}(\mathsf{ol}(\phi_m), \mathsf{ol}(\phi_h))$; $C \leftarrow\!\!\$\ \{0,1\}^c$
Return (N, C)	$S_j \leftarrow S_j \cup \{(N, H, C)\}$
	Return (N, C)
proc DEC(N, H, C) // $\mathrm{KIAE}_{\mathsf{SE},\mathtt{nt}}$	proc DEC(j, N, ϕ_h, C) // $\mathrm{KDAE}_{\mathsf{SE},\mathtt{nt}}$
If $(N, H, C) \in S$ then return $\perp$	$H \leftarrow \phi_h(K_1, \ldots, K_w)$
If $(b = 1)$ then $M \leftarrow \mathcal{D}(K, H, N, C)$	If $(N, H, C) \in S_j$ then return $\perp$
Else $M \leftarrow \perp$	If $(b = 1)$ then $M \leftarrow \mathcal{D}(K_j, N, H, C)$
If $M = \perp$ then $V \leftarrow 0$ else $V \leftarrow 1$	Else $M \leftarrow \perp$
Return V	If $M = \perp$ then $V \leftarrow 0$ else $V \leftarrow 1$
	Return V
proc FINALIZE(b') // $\mathrm{KIAE}_{\mathsf{SE},\mathtt{nt}}$	proc FINALIZE(b') // $\mathrm{KDAE}_{\mathsf{SE},\mathtt{nt}}$
Return $(b' = b)$	Return $(b' = b)$

Fig. 2. On the left is game $\mathrm{KIAE}_{\mathsf{SE},\mathtt{nt}}$ defining AE-security of encryption scheme $\mathsf{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$, where $\mathtt{nt} \in \{\mathtt{u}, \mathtt{r}\}$ indicates universal or random nonce. On the right is game $\mathrm{KDAE}_{\mathsf{SE},w,\mathtt{nt}}$ defining KDI AE-security of SE.

depends only on the length of the message and header. Formally, there is a function $\mathsf{cl}(\cdot,\cdot)$ called the ciphertextlength such that $|C| = \mathsf{cl}(|M|, |H|)$ for any C that may be output by $\mathcal{E}(\cdot, \cdot, H, M)$.

As in [46, 48], $\mathcal{D}$ takes the nonce and header as an input. (In this view, the ciphertext in standard counter-mode encryption does not incude the counter. It is up to the application to transmit nonce and header if necessary, so the "ciphertext" in practice may be more than the output of $\mathcal{E}$, but in many settings the receiver gets nonce and header in out-of-band ways.) But our treatment differs from standard ones [9] in that the nonce must be explicitly provided to $\mathcal{D}$ even when it is random. This means that, for randomized schemes, we are limited to ones that make the randomness public, but this is typically true. The restriction is only to compact and unify the presentation. Otherwise we would have needed separate games to treat universal and random nonce security.

AE SECURITY. We now define standard (neither message nor header is key-dependent) AE security for $\mathsf{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$. Consider game $\mathrm{KIAE}_{\mathsf{SE},\mathtt{nt}}$ shown on the left side of Fig. 2. Define the advantage of adversary A via $\mathbf{Adv}^{\mathtt{ae\text{-}nt}}_{\mathsf{SE}}(A) = 2\Pr[\mathrm{KIAE}^{A}_{\mathsf{SE},\mathtt{nt}} \Rightarrow \mathsf{true}] - 1$. When $\mathtt{nt} = \mathtt{u}$ the definition captures what we call universal-nonce security. (It is simply called nonce-based security in [46, 44, 48].) It is understood that in this case we only consider A that is *unique-nonce*, meaning we have $N \neq N'$ for any two ENC queries N, H, M and N', H', M'. Security

is thus required even for adversary-chosen nonces as long as no nonce is used for more than one encryption. When $\mathtt{nt} = \mathtt{r}$, the adversary-provided nonce in ENC is ignored, a random value being substituted by the game, and we have random-nonce security, in the classical spirit of randomized encryption [30, 9]. The nonce returned by ENC is redundant in the $\mathtt{u}$ case but needed in the $\mathtt{r}$ case and thus always returned for uniformity.

Historically the first definitions of security for AE had separate privacy (IND-CPA) and integrity (INT-CTXT) requirements [12, 37, 14]. Our version is a blend of the single-game formulation of [48] and INT-CTXT. Privacy is in the strong sense of indistinguishability from random, meaning ciphertexts are indistinguishable from random strings, which implies the more common LR-style [9] privacy, namely that ciphertexts of different messages are indistinguishable from each other. (A subtle point is that the length-respecting property assumed of $\mathcal{E}$ is important for this implication.) The integrity is in the fact that the adversary can't create new ciphertexts with non-$\perp$ decryptions. ("New" means not output by ENC.) Unlike [48], oracle DEC does not return decryptions but only whether or not they succeed. This simpler version is nonetheless equivalent to the original. IND-CCA is implied by this definition of AE [12, 44].

KDI SECURITY OF AE. We now extend the above along the lines of [18, 6] to provide our definition of security for AE in the presence of key-dependent inputs, considering both key-dependent messages and key-dependent headers. Consider game $\mathrm{KDAE}_{\mathsf{SE},\mathtt{nt}}$ shown on the right side of Fig. 2. Define the advantage of adversary A via $\mathbf{Adv}^{\mathtt{ae\text{-}nt}}_{\mathsf{SE}}(A) = 2\Pr[\mathrm{KDAE}^{A}_{\mathsf{SE},\mathtt{nt}} \Rightarrow \mathsf{true}] - 1$. The argument w to INITIALIZE is the number of keys; arguments ϕ_m, ϕ_h (message and header deriving functions, respectively) in the ENC, DEC queries must be functions in $\mathsf{Fun}(k, w)$; $\mathsf{ol}(\phi)$ is the output length of $\phi \in \mathsf{Fun}(k, w)$; and cl is the ciphertext length of SE. When $\mathtt{nt} = \mathtt{u}$ the definition again captures universal-nonce security. That A is unique-nonce (always assumed in this case) now means that for each $j \in [1..w]$ we have $N \neq N'$ for any two ENC queries j, N, ϕ_m, ϕ_h and j, N', ϕ'_m, ϕ'_h. When $\mathtt{nt} = \mathtt{r}$ we have random-nonce security.

Messages could be key-dependent or not, and so could headers, giving rise to four settings of interest. These are best captured by considering different classes of adversaries. For $\Phi_m, \Phi_h \subseteq \mathsf{Fun}(k, w)$ let $\mathcal{A}[\Phi_m, \Phi_h]$ be the class of all adversaries A for which ϕ_m in A's ENC queries is in Φ_m and ϕ_h in its ENC, DEC queries is in Φ_h. Let $\mathcal{A}[\mathtt{mt}, \mathtt{ht}] = \mathcal{A}[\Phi_m, \Phi_h]$ where the values of (Φ_m, Φ_h) corresponding to $(\mathtt{mt}, \mathtt{ht}) = (\mathtt{kd}, \mathtt{kd}), (\mathtt{kd}, \mathtt{ki}), (\mathtt{ki}, \mathtt{kd}), (\mathtt{ki}, \mathtt{ki})$ are, respectively, $(\mathsf{Fun}(k, w), \mathsf{Fun}(k, w))$, $(\mathsf{Fun}(k, w), \mathsf{Cns}(k, w))$, $(\mathsf{Cns}(k, w), \mathsf{Fun}(k, w))$, $(\mathsf{Cns}(k, w), \mathsf{Cns}(k, w))$. Say that $\mathsf{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ is $(\mathtt{nt}, \mathtt{mt}, \mathtt{ht})$-AE secure if $\mathbf{Adv}^{\mathtt{ae\text{-}nt}}_{\mathsf{SE}}(A)$ is negligible for all efficient $A \in \mathcal{A}[\mathtt{mt}, \mathtt{ht}]$.

Now that the header may not be known to the adversary in a DEC query, it does not know in advance whether or not $(H, N, C) \in S_j$ and it deserves to know whether rejection took place due to this or due to unsuccessful decryption. This why we do not return $\perp$ for both but rather $\perp$ for one and 0 for the other. It was to disambiguate these that we found it convenient to modify the

starting definition of AE. The issue is crucial when considering security with key-dependent headers.

In the RO model there is an additional procedure HASH representing the RO. As usual it may be invoked by the scheme algorithms and the adversary, but, importantly, also by the input-deriving functions ϕ_m, ϕ_h.

For input-deriving functions to be adversary queries it is assumed they are encoded in some way. Recall that, as per our convention, the running time of A is that of the execution of A with the game, so A pays in run time if it uses functions whose description or evaluation time is too long. In asymptotic terms, A is restricted to polynomial-time computable input-deriving functions, and their description could be set to the Turing-machine that computes them.

PASSWORDS AS KEYS. The key-generation algorithm $\mathcal{K}$ in our syntax $\mathsf{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ does not have to output random k-bit strings but could induce an arbitrary distribution, allowing us to capture passwords. The metric of interest in this case is the min-entropy $\mathbf{H}_\infty(\mathcal{K}) = -\log_2(\mathbf{GP}(\mathcal{K}))$, where the guessing probability $\mathbf{GP}(\mathcal{K})$ is defined as the maximum, over all k-bit strings K, of the probability that $K' = K$ when $K' \leftarrow\!\!\$\, \mathcal{K}$. We aim to provide security as long as the min-entropy of the key-generator is not too small.

Providing security when keys are passwords is crucial because key-dependent data is more natural and prevalent in this case. In practice, our keys are largely passwords. They may be stored on disk. Their hashes are stored on the disk by the system.

3 Impossibility Results

We rule out universal-nonce security for key-dependent messages as well as security for key-dependent headers.

3.1 Universal-nonce insecurity

Standardized schemes all achieve universal-nonce security for `ki`-messages. This is convenient because an application-setting often provides for free something that can play the role of a nonce, like a counter. It also increases resistance to misuse. We would like to maintain this type of security in the presence of key-dependent data. Unfortunately we show that this is impossible. We show that no scheme is $(\mathtt{u}, \mathtt{kd}, \mathtt{ki})$-AE secure:

Proposition 1. *Let $\mathsf{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be an encryption scheme. Then there is an efficient adversary $A \in \mathcal{A}[\mathtt{kd}, \mathtt{ki}]$ such that $\mathbf{Adv}^{\mathtt{ae\text{-}u}}_{\mathsf{SE}}(A) \geq 1/4$.*

As the proof of the above will show, the attack we present is strong in that the adversary does not just distinguish real from random encryptions but recovers the key. (A simpler attack is possible if we only want to distinguish rather than recover the key.) Also the attack works even when the nonce is a counter rather than adversary controlled. And since the adversary does not use the decryption oracle we rule out even KDM-CPA security.

We begin with some background and an overview, then prove Proposition 1, and finally show how to apply an underlying lemma to provide the first analysis of an attack in BRS [18].

Background and overview. BRS [18, Section 6] suggest an attack aimed at showing that no stateful symmetric encryption scheme is KDM-secure. For the purpose of our discussion we adapt it to an attack on universal-nonce security of an AE scheme $\mathsf{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$. Let k be the keylength of the scheme. We will use messages of length m. Let c denote the length of the resulting ciphertexts. Let $H_{\mathrm{ip}}(V, C) = V[1]C[1]+\cdots+V[c]C[c] \bmod 2$ denote the inner product modulo two of c-bit strings V, C. Let $\phi_{V,i}$ denote the message-deriving function that on input a key K returns the first m-bit message M such that $H_{\mathrm{ip}}(V, \mathcal{E}(K, i, \varepsilon, M)) = K[i]$, or 0^m if there is no such message. (Here we use i as the nonce and ε as the header.) The adversary can pick V (BRS do not say how, but the natural choice is at random), query $\phi_{V,i}$ to get (i, C), and then recover $K[i]$ as $H_{\mathrm{ip}}(V, C)$, repeating for $i = 1, \ldots, k$ to get K.

The difficulty is that $\phi_{V,i}$ must search the message space until it finds a message satisfying the condition, and it is unclear how long this will take. In asymptotic terms, this means there is no proof that the attack runs in polynomial time, meaning is a legitimate attack at all. This issue does not appear to be recognized by BRS, who provide no analysis or formal claims relating to the attack.

In order to have a polynomial time attack where the key-recovery probability is, say, a constant, one would need to show that there is a polynomial number l of trials in which the failure probability to recover a particular bit $K[i]$ of the key is $O(1/k)$. (A union bound will then give the desired result.) We did not see a direct way to show this. Certainly, for a particular i, the probability that the first message M fails to satisfy $H_{\mathrm{ip}}(V, \mathcal{E}(K, i, \varepsilon, M)) = K[i]$ is at most $1/2$, but it is not clear what is the failure probability in multiple trials because they all use the same V. The first thought that comes to mind is to modify the attack so that $\phi_{V_1,\ldots,V_l,i}$ now depends on a sequence $V_1, \ldots, V_l$ of strings, chosen independently at random by the adversary. On input the key K, the function computes the smallest j such that $H_{\mathrm{ip}}(V_j, \mathcal{E}(K, i, \varepsilon, M_j)) = K[i]$, where $M_1, M_2, \ldots, M_l$ is a fixed sequence of messages, and returns M_j. Although one can prove that this "successful" j is quickly found, the attack fails to work, since, to recover $K[i] = H_{\mathrm{ip}}(V_j, C)$ from the ciphertext $C = \mathcal{E}(K, i, \varepsilon, M_j)$, the adversary needs to know j, and it is not clear how the ciphertext is to "communicate" the value of j to the adversary.

We propose a different modification, namely to replace the inner product function with a family $H\colon \{0,1\}^s \times \{0,1\}^c \to \{0,1\}$ of pairwise independent functions. The message-deriving function $\phi_{S,i}$, on input K, will now search for M such that $H(S, \mathcal{E}(K, i, \varepsilon, M)) = K[i]$. The adversary can pick S at random, query $\phi_{S,i}$ to get (i, C), and then recover $K[i]$ as $H(S, C)$, repeating for $i = 1, \ldots, k$ to get K. We will prove that $O(k)$ trials suffice for the search to have failure probability at most $O(1/k)$ for each i, and thus that the adversary gets a constant advantage in a linear number of trials.

This strategy can be instantiated by the pairwise independent family of functions $H\colon \{0,1\}^{c+1} \times \{0,1\}^c \to \{0,1\}$ defined by $H(S,C) = H_{\mathrm{ip}}(S[1]\ldots S[c],C) + S[c+1] \bmod 2$ to get a concrete attack that is only a slight modification of the BRS one but is proven to work. Given this, the question of whether the original attack can be proven to work is perhaps moot, but we find it interesting for historical reasons. Our results would not at first appear to help to answer this because the inner product function is not pairwise independent. (For example, 0^c is mapped to 0 by all functions in the family.) But curiously, as a corollary of our proof that the attack works for the *particular* family H we just defined, we get a proof that the BRS attack works as well. This is because we show that the attack using H works for an overwhelming fraction of functions from H, and thus, with sufficient probability, even for functions drawn only from the subspace of inner-product functions. Let us now proceed to the details.

Attack and analysis. We begin with a general lemma.

Lemma 2. *Let $H\colon \{0,1\}^s \times \{0,1\}^c \to \{0,1\}$ be a family of pairwise independent hash functions. Let $C_1,\ldots,C_l \in \{0,1\}^c$ be distinct and let $T \in \{0,1\}$. Then*

$$\Pr[\,\forall j \,:\, H(S,C_j) \neq T\,] \;\leq\; \frac{1}{l}$$

where the probability is over a random choice of S from $\{0,1\}^s$.

Proof (Lemma 2). For each $j \in \{1,\ldots,l\}$ define $X_j\colon \{0,1\}^s \to \{0,1\}$ to take value 1 on input S if $H(S,C_j) = T$ and 0 otherwise. Regard $X_1,\ldots,X_j$ as random variables over the random choice of S from $\{0,1\}^s$. Let $X = X_1 + \cdots + X_l$ and let $\mu = \mathbf{E}[X]$. By Chebyshev's inequality, the probability above is

$$\Pr[\,X = 0\,] \;\leq\; \Pr[\,|X-\mu| \geq \mu\,] \;\leq\; \frac{\mathbf{Var}[X]}{\mu^2}\,.$$

Since H is pairwise independent, so are $X_1,\ldots,X_l$ and hence $\mathbf{Var}[X] = \mathbf{Var}[X_1] + \cdots + \mathbf{Var}[X_l]$. But for each j we have $\mathbf{E}[X_j] = 1/2$ and $\mathbf{Var}[X_j] = 1/4$, so $\mu = l/2$ and $\mathbf{Var}[X] = l/4$. Thus the above is at most $(l/4)/(l/2)^2 = 1/l$ as desired. □

We now use this to prove Proposition 1.

Proof (Proposition 1). Let k be the keylength, n the noncelength and cl the ciphertextlength of SE. Let $l = 4k$. Let $\mathsf{NumToStr}(j)$ denote a representation of integer $j \in \{0,\ldots,l\}$ as a string of length exactly $m = \lceil \log_2(l+1) \rceil$ bits. Let $H\colon \{0,1\}^s \times \{0,1\}^{\mathsf{cl}(m,0)} \to \{0,1\}$ denote a family of pairwise independent hash functions with s-bit keys. We construct an adversary B that recovers the target key with probability at least 3/4 when playing the real game, meaning game $\mathrm{KDAE}_{\mathsf{SE},\mathsf{u}}$ with challenge bit $b = 1$. From B it is easy to build A achieving advantage at least 1/4. Below we depict B and also define the message-deriving functions it uses. Nonces are given as integers and assumed encoded as n-bit strings:

Adversary B	Function $\phi_{\boldsymbol{m},S,i}(K)$

INITIALIZE(1)
For $j = 1, \ldots, l$ do
  $\boldsymbol{m}[j] \leftarrow \mathsf{NumToStr}(j)$
$S \leftarrow\!\!\$\ \{0,1\}^s$
For $i = 1, \ldots, k$ do
  $(i, C) \leftarrow\!\!\$\ \text{ENC}(1, i, \phi_\varepsilon, \phi_{\boldsymbol{m},S,i})$; $L[i] \leftarrow H(S, C)$
Return L
	$M \leftarrow \mathsf{NumToStr}(0)$
	For $j = 1, \ldots, l$ do
	  $C_j \leftarrow \mathcal{E}(K, i, \varepsilon, \boldsymbol{m}[j])$
	  If $H(S, C_j) = K[i]$ then
	  $M \leftarrow \boldsymbol{m}[j]$
	Return M

Above $\boldsymbol{m}$ is a l-vector over $\{0,1\}^m$ and ϕ_ε is the constant function that returns the empty string on every input. In its first step, B initializes the game to play with $w = 1$, meaning a single target key. Function $\phi_{\boldsymbol{m},S,i}(K)$ returns a message from whose encryption under nonce i and empty header one can recover bit i of the key by encoding this bit as the result of $H(S, \cdot)$ on the ciphertext. For the analysis, Lemma 2 says that for each i, adversary B fails to recover $K[i]$ with probability at most $1/4k$. By the union bound B fails to recover K with probability at most $1/4$. □

ANALYSIS OF THE BRS ATTACK. As a corollary of Lemma 2 we not only show that the inner-product function works but that it is worse only by a factor of two:

Lemma 3. *Let* $H_{\text{ip}}\colon \{0,1\}^c \times \{0,1\}^c \to \{0,1\}$ *be defined by* $H_{\text{ip}}(V, C) = V[1]C[1] + \cdots + V[c]C[c] \bmod 2$. *Let* $C_1, \ldots, C_l \in \{0,1\}^c$ *be distinct and let* $T \in \{0,1\}$. *Then*

$$\Pr\left[\,\forall j \,:\, H_{\text{ip}}(V, C_j) \neq T\,\right] \;\leq\; \frac{2}{l} \tag{1}$$

where the probability is over a random choice of V *from* $\{0,1\}^c$.

Proof (Lemma 3). Define $H\colon \{0,1\}^{c+1} \times \{0,1\}^c \to \{0,1\}$ by

$$H(S, C) = H_{\text{ip}}(S[1] \ldots S[c], C) + S[c+1] \bmod 2\ .$$

This family of functions is pairwise independent. Let G be the set of all $S \in \{0,1\}^{c+1}$ such that $H(S, C_j) = T$ for some j. For $b \in \{0,1\}$ let G_b be the set of all $S \in G$ with $S[c+1] = b$. Let $\epsilon = 1/l$. Lemma 2 says that $|G| \geq (1-\epsilon)2^{c+1}$. But $G = G_0 \cup G_1$ and G_0, G_1 are disjoint so

$$|G_0| = |G| - |G_1| \;\geq\; |G| - 2^c \;\geq\; (1-\epsilon)2^{c+1} - 2^c \;=\; (1-2\epsilon)2^c\ .$$

To conclude we note that the probability on the left of Equation (1) equals $1 - |G_0|/2^c$. □

With this in hand, one can substitute H by H_{ip} in the proof of Lemma 1. By also doubling the value of l, the analysis goes through and shows that the BRS attack terminates in a linear number of trials and achieves a constant advantage.

3.2 Header insecurity

We would like to use schemes in such a way that headers are not key-dependent but it may not be under our control. Applications may create headers based on data present on the system in a way that results in their depending on the key. We would thus prefer to maintain security in the presence of key-dependent headers. We show that this, too, is impossible, even when messages are key-independent. For both $\mathtt{nt} = \mathtt{u}$ and $\mathtt{nt} = \mathtt{r}$, we present attacks showing no scheme is $(\mathtt{nt}, \mathtt{ki}, \mathtt{kd})$-secure.

Proposition 4. *Let* $\mathsf{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ *be an encryption scheme. Then for any* $\mathtt{nt} \in \{\mathtt{u}, \mathtt{r}\}$ *there is an efficient adversary* $A \in \mathcal{A}[\mathtt{ki}, \mathtt{kd}]$ *such that* $\mathbf{Adv}^{\mathtt{ae\text{-}nt}}_{\mathsf{SE}}(A) \geq 1/2$.

Proof (Proposition 4). Let k be the keylength of SE. Again, we present an adversary B that recovers the key with probability 1, from which A is easily built. Below we depict B and also define the message-deriving functions it uses. Nonces are given as integers and assumed encoded as n-bit strings:

Adversary B	

INITIALIZE(1)
For $i = 1, \ldots, k$ do
  $(N_i, C_i) \leftarrow\!\!\$\ \textsc{Enc}(1, i, \mathsf{bit}_i, \phi_0)$; $V_i \leftarrow \textsc{Dec}(1, N_i, \phi_0, C_i)$
  If $V_i = \perp$ then $L[i] \leftarrow 0$ else $L[i] \leftarrow 1$
Return L
	Function $\mathsf{bit}_i(K)$
	Return $K[i]$

Here ϕ_c denotes the constant function that returns $c \in \{0, 1\}$. The header computed and used by the game in response to the i-th ENC query is $K[i]$. The header computed and used by the game in response to the i-th DEC query is 0. Thus, DEC will return $\perp$ if $K[i] = 0$. Otherwise, it will most likely return 0 because the headers don't match, although it might return 1, but in either case we have learned that $K[i] = 1$.

The attack has been written so that it applies in both the universal and random nonce cases. In the first case we will have $N_i = i$. In the second case, N_i will be a random number independent of i chosen by the game.

3.3 Remarks

The message-deriving functions used by the adversary in the proof of Proposition 1 invoke the encryption algorithm, which is legitimate since any efficient function is allowed. Having encryption depend on a RO will not avoid the attack because the message-deriving functions are allowed to call the RO and can continue to compute encryptions. (In an instantiation the RO will be a hash function and the system may apply it to the key to get data that is later encrypted.)

We do not suggest that precisely these attacks may be mounted in practice. (The message-deriving functions in our attacks are contrived.) However, our attacks rule out the possibility of a proof of security and thus there may exist other, more practical attacks. Indeed, the history of cryptography shows that once an attack is uncovered, better and more practical ones often follow.

4 The RHtE transform and its security

We describe our RHtE (Randomized Hash then Encrypt) transform and prove that it endows the base scheme to which it is applied with $(\mathtt{r}, \mathtt{kd}, \mathtt{ki})$-AE security.

THE TRANSFORM. Given a base symmetric encryption scheme $\mathsf{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$, a key-generation algorithm $\mathcal{L}$ returning l-bit strings, and an integer parameter r representing the length of the random seed used in the key-hashing, the RHtE transform returns a new symmetric encryption scheme $\overline{\mathsf{SE}} = \mathsf{RHtE}[\mathsf{SE}, \mathcal{L}, r] = (\mathcal{L}, \overline{\mathcal{E}}, \overline{\mathcal{D}})$. It has $\mathcal{L}$ as its key-generation algorithm, keylength l, noncelength r and the same ciphertextlength as the base scheme. Its encryption and decryption algorithms are defined as follows, where HASH: $\{0,1\}^{r+l} \to \{0,1\}^k$ is a RO, $L \in \{0,1\}^l$ is the key, $R \in \{0,1\}^r$ is the nonce (which in the security game will be random), H is the header and M is the message:

Algorithm $\overline{\mathcal{E}}(L, R, H, M)$	Algorithm $\overline{\mathcal{D}}(L, R, H, C)$
$K \leftarrow \text{HASH}(R \,\|\, L)$; $C \leftarrow \mathcal{E}(K, H, M)$	$K \leftarrow \text{HASH}(R \,\|\, L)$; $M \leftarrow \mathcal{D}(K, H, C)$
Return C	Return M

The base scheme $\mathsf{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ is assumed to achieve standard $(\mathtt{nt}, \mathtt{ki}, \mathtt{ki})$-AE security, with $\mathtt{nt}$ being either $\mathtt{u}$ or $\mathtt{r}$. It is assumed to be a standard (as opposed to RO) model scheme. This is not a restriction because for the type of security we assume of it (no key-dependent data) there is no need to use a RO and none of the standardized, in use schemes do, and in any case the assumption is only for simplicity. We are not concerned with keys of the base scheme being passwords because, in standard schemes, they aren't. (Most of the time the key is an AES key.) So it is assumed that $\mathcal{K}$ returns random strings of length k. We only require one-time security of the base scheme. Accordingly we assume it is nonceless and deterministic and drop the nonce input above for both encryption and decryption. One can obtain such a scheme from standard ones by fixing a single, public nonce and hardwiring it into the algorithm. The repeated use of the nonce causes no problems since the key K is different on each encryption.

We want the constructed scheme $\overline{\mathsf{SE}}$ to provide security not only when its keys are full-fledged cryptographic ones but also when they are passwords. Hence we view as given an (arbitrary) key-generation algorithm $\mathcal{L}$ returning l-bit strings under some arbitrary distribution, and design $\overline{\mathsf{SE}}$ to have $\mathcal{L}$ as its key-generation algorithm.

The ciphertext returned is a ciphertext of the base scheme but this is deceptive since in practice R will have to be transmitted too to enable decryption. Nonetheless, in common usage, there will be no bandwidth overhead. This is because we must compare to a standard use of the base scheme where it too uses and transmits a nonce. We have saved this space by fixing this nonce and can use it for R. However, if we are in a mode where the base scheme gets the nonce out-of-band, we have r bits of bandwidth overhead. The computational overhead is independent of the message size. Implementations with base schemes CCM, EAX and GCM (see Section 5) show that for the first the slowdown is 11% for 5KB messaegs and only 1% for 50KB messages.

The BRS scheme [18] is purely RO-based, and one needs ROs with outputs of length equal to the length of the message. In our scheme the RO is used only for key-derivation and its output length is independent of the length of the message to be encrypted. In this sense, the reliance on ROs is reduced.

Security of RHtE. The following theorem says that if the base scheme is secure for key-independent headers and messages then the constructed scheme is random-nonce secure for key-dependent messages and key-independent headers.

Theorem 5. *Let $\mathsf{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be a base symmetric encryption scheme as above. Let $\mathcal{L}$ be a key-generation algorithm with keylength l and let r be a positive integer. Let $\overline{\mathsf{SE}} = \mathsf{RHtE}[\mathsf{SE}, \mathcal{L}, r]$ be the RO model symmetric encryption scheme associated to $\mathsf{SE}, \mathcal{L}, r$ as above. Let $A \in \mathcal{A}[\mathtt{kd}, \mathtt{ki}]$ be an adversary making $\mathsf{q_e}$ Enc queries, $\mathsf{q_d}$ Dec queries and $\mathsf{q_h}$ Hash queries, and let $w \leq 2^{\mathbf{H}_\infty(\mathcal{L})-1}$ be the number of keys, meaning the argument of A's Initialize query. Then there is an adversary D such that*

$$\mathbf{Adv}^{\mathrm{ae\text{-}r}}_{\overline{\mathsf{SE}}}(A) \leq (24\mathsf{q_e}^2 + 2\mathsf{q_d}) \cdot \mathbf{Adv}^{\mathrm{ae}}_{\mathsf{SE}}(D) + \frac{8w\mathsf{q_e}\mathsf{q_h} + 2w(w-1)\mathsf{q_e}}{2^{\mathbf{H}_\infty(\mathcal{L})}} + \frac{2\mathsf{q_e}(\mathsf{q_h} + 2\mathsf{q_e}w)}{2^r} \quad (2)$$

Adversary D makes only one Enc *query and has the same number of* Dec *queries and the same time complexity as A.* ▌

We have omitted the $\mathtt{nt}$ superscript in the advantage of D because SE is nonceless. That only one-time security is required of SE is reflected in the fact that D makes only one Enc query. We remark that the bound in Theorem 5 does not appear to be tight. It is an interesting open problem to either provide a proof with a better bound or an alternative scheme for which a tight bound can be proved.

Proof overview. As we noted in Section 1 the proof is surprisingly involved because message-deriving functions are allowed to query the RO and because the assumed security of the base scheme must be invoked in multiple, interrelated ways in different parts of the argument, leading to two hybrids in opposite directions, one, unusually, with steps that are differently weighted.

Assume for simplicity that $w = 1$, meaning there is a single target key, denoted L. Also assume A makes no Dec queries. Denote by $\phi_1, \ldots, \phi_{\mathsf{q_e}}$ the message-deriving functions in its Enc queries and ignore the corresponding headers. Picking index g at random we set up a hybrid in which the i-th Enc query ϕ_i is answered by encrypting message $\phi_i(L)$ under L as in the real game if $i < g$ and answered at random if $i > g$, the g-th query toggling between real and random to play the role of the challenge for an adversary B against the base scheme. Let $R_1, \ldots, R_{\mathsf{q_e}}$ denote the random nonces chosen by the game. The reduction B cannot answer hash oracle query $R_g \| L$ because the reply is its target key so a bad event is flagged if A either makes this query directly, or indirectly via a message-deriving function. But once query g has been answered, A has R_g and

Hash	Scheme	RHtE Relative Running Time			
		KeySetup	5KB	50KB	500KB
SHA256	CCM	2.73	1.11	1.01	1.00
	EAX	1.94	1.10	1.01	1.00
	GCM-2k	1.66	1.10	1.02	1.00
	GCM-64k	1.19	1.09	1.02	1.00

Fig. 3. Table showing relative slowdown of RHtE with SHA256 in Crypto++ for common AE schemes and different message sizes. KeySetup is the relative slowdown in the keysetup phase alone. GCM-2k and GCM-64k correspond to GCM implemented with tables of corresponding size.

thus for queries $i > g$, nothing can prevent ϕ_i from querying $R_g \| L$ to the RO, and how are these queries to be answered by B? Crucial to this was doing the hybrid top to bottom, meaning first real then random rather than the other way round. This enables us to avoid evaluating ϕ_i on L for post-challenge queries, so that its RO queries do not need to be answered at all. This leaves the possibility that A directly makes hash query $R_g \| L$ after it gets R_g. Intuitively this is unlikely because A does not know L. The subtle point is that this relies on the assumed security of the base scheme and hence must be proven by reduction. However, doing such a reduction means another hybrid and seems to simply shunt the difficulty to another query. To get around this circularity, we do the second hybrid in the opposite direction and also with different "weights" on the different steps. A full proof can be found in [10].

5 Implementation results

We recall that RHtE works on an existing AE scheme and a hash function. We ran RHtE with common AE schemes like CCM, EAX and GCM (with tables of 2k and 64k entries) to measure the slowdown relative to the original schemes, using a truncated version of SHA256 as the hash function and setting $l = r = k = 128$. We ran these tests using Crypto++ [26], a standard cryptography library. The measurements in Fig. 3 correspond to a Intel Core i5 M460 64-bit CPU running at 2.53 GHz with code compiled using g++ -O3 for data sizes small enough to fit in the level 2 cache. For our purposes, the relative performance of these routines is of more importance. From Fig. 3, we can observe that even at modest message sizes of around 50KB, the slowdown due to RHtE is no more than 1%. Futhermore, if algorithms like GCM are implemented with large tables and in turn a lot of precomputation in the key-setup phase, the RHtE overhead is even less noticeable.

Acknowledgments

The authors are supported in part by NSF grants CNS-0904380 and CCF-0915675. We thank the Crypto 2011 reviewers for their comments.

References

1. P. Abeni, L. Bello, and M. Bertacchini. Exploiting DSA-1571: How to break PFS in SSL with EDH, July 2008. `http://www.lucianobello.com.ar/exploiting_DSA-1571/index.html`.
2. T. Acar, M. Belenkiy, M. Bellare, and D. Cash. Cryptographic agility and its relation to circular encryption. In H. Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 403–422. Springer, May 2010.
3. B. Applebaum. Key-dependent message security: Generic amplification and completeness. In K. G. Paterson, editor, *EUROCRYPT 2011*, volume 6632 of *LNCS*, pages 527–546. Springer, May 2011.
4. B. Applebaum, D. Cash, C. Peikert, and A. Sahai. Fast cryptographic primitives and circular-secure encryption based on hard learning problems. In S. Halevi, editor, *CRYPTO 2009*, volume 5677 of *LNCS*, pages 595–618. Springer, Aug. 2009.
5. M. Backes, M. Dürmuth, and D. Unruh. OAEP is secure under key-dependent messages. In J. Pieprzyk, editor, *ASIACRYPT 2008*, volume 5350 of *LNCS*, pages 506–523. Springer, Dec. 2008.
6. M. Backes, B. Pfitzmann, and A. Scedrov. Key-dependent message security under active attacks - brsim/uc-soundness of dolev-yao-style encryption with key cycles. *Journal of Computer Security*, 16(5):497–530, 2008.
7. B. Barak, I. Haitner, D. Hofheinz, and Y. Ishai. Bounded key-dependent message security. In H. Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 423–444. Springer, May 2010.
8. M. Bellare, Z. Brakerski, M. Naor, T. Ristenpart, G. Segev, H. Shacham, and S. Yilek. Hedged public-key encryption: How to protect against bad randomness. In M. Matsui, editor, *ASIACRYPT 2009*, volume 5912 of *LNCS*, pages 232–249. Springer, Dec. 2009.
9. M. Bellare, A. Desai, E. Jokipii, and P. Rogaway. A concrete security treatment of symmetric encryption. In *38th FOCS*, pages 394–403. IEEE Computer Society Press, Oct. 1997.
10. M. Bellare and S. Keelveedhi. Authenticated and misuse-resistant encryption of key-dependent data. Cryptology ePrint Archive, Report 2011/269, 2011. Full version of this paper, `http://eprint.iacr.org/`.
11. M. Bellare and T. Kohno. A theoretical treatment of related-key attacks: RKA-PRPs, RKA-PRFs, and applications. In E. Biham, editor, *EUROCRYPT 2003*, volume 2656 of *LNCS*, pages 491–506. Springer, May 2003.
12. M. Bellare and C. Namprempre. Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. In T. Okamoto, editor, *ASIACRYPT 2000*, volume 1976 of *LNCS*, pages 531–545. Springer, Dec. 2000.
13. M. Bellare and P. Rogaway. Optimal asymmetric encryption. In A. D. Santis, editor, *EUROCRYPT'94*, volume 950 of *LNCS*, pages 92–111. Springer, May 1994.
14. M. Bellare and P. Rogaway. Encode-then-encipher encryption: How to exploit nonces or redundancy in plaintexts for efficient cryptography. In T. Okamoto, editor, *ASIACRYPT 2000*, volume 1976 of *LNCS*, pages 317–330. Springer, Dec. 2000.
15. M. Bellare and P. Rogaway. The security of triple encryption and a framework for code-based game-playing proofs. In S. Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 409–426. Springer, May / June 2006.
16. M. Bellare, P. Rogaway, and D. Wagner. The EAX mode of operation. In B. K. Roy and W. Meier, editors, *FSE 2004*, volume 3017 of *LNCS*, pages 389–407. Springer, Feb. 2004.

17. N. Bitansky and R. Canetti. On strong simulation and composable point obfuscation. In T. Rabin, editor, *CRYPTO 2010*, volume 6223 of *LNCS*, pages 520–537. Springer, Aug. 2010.
18. J. Black, P. Rogaway, and T. Shrimpton. Encryption-scheme security in the presence of key-dependent messages. In K. Nyberg and H. M. Heys, editors, *SAC 2002*, volume 2595 of *LNCS*, pages 62–75. Springer, Aug. 2003.
19. D. Boneh, S. Halevi, M. Hamburg, and R. Ostrovsky. Circular-secure encryption from decision diffie-hellman. In D. Wagner, editor, *CRYPTO 2008*, volume 5157 of *LNCS*, pages 108–125. Springer, Aug. 2008.
20. Z. Brakerski and S. Goldwasser. Circular and leakage resilient public-key encryption under subgroup indistinguishability - (or: Quadratic residuosity strikes back). In T. Rabin, editor, *CRYPTO 2010*, volume 6223 of *LNCS*, pages 1–20. Springer, Aug. 2010.
21. Z. Brakerski, S. Goldwasser, and Y. T. Kalai. Black-box circular-secure encryption beyond affine functions. In Y. Ishai, editor, *TCC 2011*, volume 6597 of *LNCS*, pages 201–218. Springer, Mar. 2011.
22. D. R. Brown. A weak randomizer attack on RSA-OAEP with e=3. IACR ePrint Archive, 2005.
23. J. Camenisch, N. Chandran, and V. Shoup. A public key encryption scheme secure against key dependent chosen plaintext and adaptive chosen ciphertext attacks. In A. Joux, editor, *EUROCRYPT 2009*, volume 5479 of *LNCS*, pages 351–368. Springer, Apr. 2009.
24. J. Camenisch and A. Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In B. Pfitzmann, editor, *EUROCRYPT 2001*, volume 2045 of *LNCS*, pages 93–118. Springer, May 2001.
25. R. Canetti, Y. T. Kalai, M. Varia, and D. Wichs. On symmetric encryption and point obfuscation. In D. Micciancio, editor, *TCC 2010*, volume 5978 of *LNCS*, pages 52–71. Springer, Feb. 2010.
26. W. Dai. Crypto++ library. `http://www.cryptopp.com`.
27. L. Dorrendorf, Z. Gutterman, and B. Pinkas. Cryptanalysis of the windows random number generator. In P. Ning, S. D. C. di Vimercati, and P. F. Syverson, editors, *ACM CCS 07*, pages 476–485. ACM Press, Oct. 2007.
28. E. Fujisaki, T. Okamoto, D. Pointcheval, and J. Stern. RSA-OAEP is secure under the RSA assumption. *Journal of Cryptology*, 17(2):81–104, Mar. 2004.
29. I. Goldberg and D. Wagner. Randomness in the Netscape browser. Dr. Dobb's Journal, January 1996.
30. S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270–299, 1984.
31. M. González. *Cryptography in the Presence of Key Dependent Messages.* PhD thesis, Florida Atlantic University, 2009.
32. M. Green and S. Hohenberger. CPA and CCA-secure encryption systems that are not 2-circular secure. Cryptology ePrint Archive, Report 2010/144, 2010. `http://eprint.iacr.org/`.
33. Z. Gutterman and D. Malkhi. Hold your sessions: An attack on Java session-id generation. In A. Menezes, editor, *CT-RSA 2005*, volume 3376 of *LNCS*, pages 44–57. Springer, Feb. 2005.
34. I. Haitner and T. Holenstein. On the (im)possibility of key dependent encryption. In O. Reingold, editor, *TCC 2009*, volume 5444 of *LNCS*, pages 202–219. Springer, Mar. 2009.

35. S. Halevi and H. Krawczyk. Security under key-dependent inputs. In P. Ning, S. D. C. di Vimercati, and P. F. Syverson, editors, *ACM CCS 07*, pages 466–475. ACM Press, Oct. 2007.
36. D. Hofheinz and D. Unruh. Towards key-dependent message security in the standard model. In N. P. Smart, editor, *EUROCRYPT 2008*, volume 4965 of *LNCS*, pages 108–126. Springer, Apr. 2008.
37. J. Katz and M. Yung. Unforgeable encryption and chosen ciphertext secure modes of operation. In B. Schneier, editor, *FSE 2000*, volume 1978 of *LNCS*, pages 284–299. Springer, Apr. 2000.
38. T. Malkin, I. Teranishi, and M. Yung. Efficient circuit-size independent public key encryption with KDM security. In K. G. Paterson, editor, *EUROCRYPT 2011*, volume 6632 of *LNCS*, pages 507–526. Springer, May 2011.
39. D. A. McGrew and J. Viega. The security and performance of the Galois/counter mode (gcm) of operation. In A. Canteaut and K. Viswanathan, editors, *INDOCRYPT 2004*, volume 3348 of *LNCS*, pages 343–355. Springer, Dec. 2004.
40. M. G. Muñiz and R. Steinwandt. Security of signature schemes in the presence of key-dependent messages. *Tatra Mt. Math. Publ.*, 47:15–29, 2010.
41. M. Mueller. Debian OpenSSL predictable PRNG bruteforce SSH exploit, May 2008. `http://milw0rm.com/exploits/5622`.
42. K. Ouafi and S. Vaudenay. Smashing SQUASH-0. In A. Joux, editor, *EUROCRYPT 2009*, volume 5479 of *LNCS*, pages 300–312. Springer, Apr. 2009.
43. K. G. Paterson and G. J. Watson. Plaintext-dependent decryption: A formal security treatment of SSH-CTR. In H. Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 345–361. Springer, May 2010.
44. P. Rogaway. Authenticated-encryption with associated-data. In V. Atluri, editor, *ACM CCS 02*, pages 98–107. ACM Press, Nov. 2002.
45. P. Rogaway. Efficient instantiations of tweakable blockciphers and refinements to modes OCB and PMAC. In P. J. Lee, editor, *ASIACRYPT 2004*, volume 3329 of *LNCS*, pages 16–31. Springer, Dec. 2004.
46. P. Rogaway. Nonce-based symmetric encryption. In B. K. Roy and W. Meier, editors, *FSE 2004*, volume 3017 of *LNCS*, pages 348–359. Springer, Feb. 2004.
47. P. Rogaway, M. Bellare, J. Black, and T. Krovetz. OCB: A block-cipher mode of operation for efficient authenticated encryption. In *ACM CCS 01*, pages 196–205. ACM Press, Nov. 2001.
48. P. Rogaway and T. Shrimpton. A provable-security treatment of the key-wrap problem. In S. Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 373–390. Springer, May / June 2006.
49. D. Whiting, R. Housley, and N. Ferguson. Counter with CBC-MAC (CCM). Undated manuscript. Submission to NIST, available from their web page, June 2002.
50. D. Whiting, R. Housley, and N. Ferguson. Counter with CBC-MAC (CCM). RFC 3610 (Informational), Sept. 2003.
51. S. Yilek, E. Rescorla, H. Shacham, B. Enright, and S. Savage. When private keys are public: Results from the 2008 Debian OpenSSL vulnerability. In *IMC*. ACM, 2009.