A preliminary version of this paper appears in the proceedings of Crypto 2026. This is the full version.

UCX is All You Need: A Universal Transform for Committing Authenticated Encryption

Mihir Bellare[1] Rishabh Ranjan[2] Nujud Senan[3] Basel Alomair[4]

June 2026

Abstract

Emerging attacks and applications have motivated the development of transforms that turn a given AE scheme into a committing AE (cAE) one. We give a new transform called **UCX** with the following attributes: It does not require the starting scheme to be tag based, works for schemes in the broad AE5 framework rather than the limited AE1 one, and preserves both UNAE (Unique Nonce AE) and MRAE (Misuse Resistant AE) security. No prior transform is "universal" in the sense of having the combination of all these properties. The use of **UCX** in place of prior, limited transforms reduces the risk of error and failure in the real world, where choices may be made by application developers and hidden in software libraries. The committing security of **UCX** is shown in the ideal-cipher model, and its AE5-security in the standard model. To design **UCX**, we introduce and build a new primitive, that we call a Tweakable Committing Concealer, and that may be of independent interest.

[1] Department of Computer Science & Engineering, University of California San Diego, USA. Email: mbellare@ucsd.edu. URL: cseweb.ucsd.edu/~mihir/. Supported in part by NSF grant CNS-2154272 and KACST.

[2] Department of Computer Science & Engineering, University of California San Diego, USA. Email: riranjan@ucsd.edu. URL: cseweb.ucsd.edu/~riranjan/. Supported in part by NSF grant CNS-2154272 and KACST.

[3] King Abdulaziz City for Science and Technology. Email: NAlyami@mcit.gov.sa.

[4] King Abdulaziz City for Science and Technology; University of Washington Seattle; HUMAIN. Email: alomair@kacst.edu.sa.

Contents

1 Introduction **3**

2 Preliminaries **8**

3 AE Framework **10**

4 Tweakable committing concealers and HER **12**

5 The UCX universal committing transform **19**

References **26**

A CIPH-COLL attack on GCM **29**

1 Introduction

AE and cAE. Modern symmetric encryption is nonce-based. The encryption algorithm $\mathsf{SE.Enc}$ of a scheme SE takes key K, nonce N, associated data A and message M to deterministically return a ciphertext $C \leftarrow \mathsf{SE.Enc}(K, N, A, M)$. AE security asks for privacy of the message and authenticity of both the message and the associated data. AE schemes are widely standardized and used, including in TLS.

But now we are seeing an increasing demand for AE schemes that are also committing. Different definitions given for this goal [24, 1, 25, 22, 2, 4, 14, 34] have converged to one strong and simple one from [4, 14], namely that the function $\mathsf{SE.Enc}$ is collision resistant. That is, it is hard to find distinct tuples $(K_1, N_1, A_1, M_1), (K_2, N_2, A_2, M_2)$ such that $\mathsf{SE.Enc}(K_1, N_1, A_1, M_1) = \mathsf{SE.Enc}(K_2, N_2, A_2, M_2)$. We will refer to this simply as *committing*, abbreviated CMT (other names are CMT-4 [4], CMT-XX [14] and context commitment [34]). Absence of CMT security in AE schemes has lead to practical attacks in several application settings including abuse reporting in encrypted messaging [17, 25], password-based encryption and key-exchange [31], anonymous public key encryption [31], key rotation [2] and envelope encryption [2]. Unfortunately, a sequence of attacks [31, 2, 25, 34, 14] also shows that popular AE schemes including GCM, $\mathsf{ChaCha20/Poly1305}$, OCB [38], CCM [41], EAX [11] and SIV [39] are all CMT-insecure. This has lead to the quest for new AE schemes that are CMT-secure (cAE).

The most attractive and popular way to achieve cAE is via generic transforms. A *committing transform* $\mathbf{CT}$ takes an encryption scheme SE and, possibly with the aid of an auxiliary primitive, returns an encryption scheme $\overline{\mathsf{SE}}$ such that two things hold: (1) AE-security is preserved: If SE is AE-secure, so is $\overline{\mathsf{SE}}$, but additionally (2) CMT-security is added: $\overline{\mathsf{SE}}$ is CMT-secure. Prior work has given several such transforms [4, 14, 5, 15, 13]. Moreover these are quite efficient. So what's the problem?

The quest for universality. The problem is that all current transforms are limited. They apply (or are shown to work) only for a sub-class of AE schemes, not all of them.

Stepping back, above we spoke of "AE schemes" as though they were one, monolithic class. The truth is the opposite. Under the lid there is in fact a cornucopia of types of AE schemes. We will discuss this in more depth in a bit, but briefly, there are structural differences (schemes can be tag based or not), different genres of AE (AE1 [37], AE2 [8], AE3 [12], AE5 [36]), and, within each genre, different security goals (UNAE [37] versus MRAE [39]). With choices in each dimension, one emerges with a large number of sub-classes of AE schemes. Now the problem is that each known transform $\mathbf{CT}$ is only shown to work for AE schemes SE in some sub-class $\mathcal{AE}_{\mathbf{CT}}$ of AE schemes, meaning it requires (assumes) that its given scheme SE is in $\mathcal{AE}_{\mathbf{CT}}$, returning a scheme $\overline{\mathsf{SE}}$ also in $\mathcal{AE}_{\mathbf{CT}}$ but additionally CMT-secure. The sub-class can differ from transform to transform, but in no case is it all AE schemes. For example, the sub-class for CTX [14] is tag-based, UNAE-secure AE1 schemes, meaning SE must obey this restriction for the transform to work.

Prior work [13, 15] has pointed out this limitation and asked whether it is possible to have a transform that is "universal" in the sense that it applies to as wide a class of AE schemes as possible; to be specific, say all the types we mentioned above. Towards an answer, they were able to give transforms that applied to larger sub-classes of AE schemes than prior transforms, but they still do not apply to the full class of all AE schemes, meaning stopped short of being fully universal. Our work fills this gap with a fully universal transform we call $\mathbf{UCX}$. It is moreover efficient as well.

Below, we will say in more detail exactly what we mean by "universal," discuss how known transforms measure up to this goal, and discuss our transform. But first one must ask about

motivation, namely why is universality desirable? We think it is of both theoretical and practical interest. On the theoretical side, it provides a conceptually clean, single solution to the problem of adding CMT security to AE. On the practical side, it simplifies application and reduces the risk of error. The choice of transform is often made by an application designer or developer. They would have some AE scheme SE in their library, and want to add CMT-security to it. Making a sound choice of transform $\mathbf{CT}$ requires understanding to what sub-class $\mathcal{AE}_{\mathbf{CT}}$ of AE schemes the transform applies and determining if SE falls in this class, which requires expertise they may not have. Legacy can also bring error. Suppose, in a certain corporation or software package, a certain choice $\mathbf{CT}$ of transform had been made and implemented for a scheme $\mathsf{SE} \in \mathcal{AE}_{\mathbf{CT}}$ to which it indeed, applies. Now suppose SE is swapped out for a new scheme. Then the same transform may automatically end up being used, even if the new scheme is not in $\mathcal{AE}_{\mathbf{CT}}$, meaning the transform does not apply.

UNIVERSALITY. We first want to define universality more precisely. Let $\mathbf{CT}$ be a transform, taking an AE scheme SE and returning a cAE scheme $\overline{\mathsf{SE}}$. We say it is *universal* under the following conditions.

SE is not required to be tag based. An AE scheme is *tag based* if its ciphertext $C = C^* \| T$ can be split into a ciphertext core C^* and a tag T such that decryption works in a very specific way: It computes from C^* a candidate tag T^* and message $M \neq \perp$, returning M if $T^* = T$ and $\perp$ otherwise. Some AE schemes, such as those built via Encrypt-then-MAC [7, 35], are tag based, but many important AE schemes are *not* tag based. In particular schemes built via the Encode-then-Encipher paradigm [9], such as AEZ [27], are not tag based. Neither are schemes built using the SIV method [39], such as AES-GCM-SIV [26]. But transforms like CTX [14], CTY [5] and XtH [33] are restricted to tag based schemes, meaning only operate and work if the starting scheme SE is tag based. As a first requirement for universality, we want to drop this restriction; the transform should work whether or not the base scheme SE is tag based.

SE can be any AE5 scheme. In the classical form of AE, called AE1 or AEAD [37], decryption gets the nonce as input, requiring the latter to be sent in the clear. This results in numerous weaknesses [8]. First, nonces can contain sensitive information, such as device identifiers, that are now revealed. Second, and more subtle, there are valid and useful choices of nonces, like hashes of the message, that if used with low-entropy messages, allow message recovery given the ciphertext and nonce, so that AE1 actually fails to fulfill its core goal of message privacy [8]. A remedy is AE2 [8], where decryption does not get the nonce, so that it is not sent in the clear and full security of the nonce and message is provided for all nonce choices. But consider a nonce N consisting of a sensitive device-id, plus a random number that can be sent in the clear. Neither AE1 nor AE2 capture this well. The broader AE3 [12] and AE5 [36] frameworks, however, do. Here the nonce $N = NP \| NS$ is explicitly divided into a public part NP and a secret part NS which the definitions treat appropriately. But most current transforms —the exceptions are CTY and XtH, but they assume tag-based SE— are only proven to work when SE is an AE1 scheme. Accordingly, the second requirement for universality of the transform $\mathbf{CT}$ is that the starting scheme SE can be AE5, not just AE1. Note that AE1, AE2 are special cases of AE5 and hence are automatically covered by this generalization.

Readers (including reviewers) sometimes question targeting AE5, asking for evidence that it arises in applications. The first answer is yes. Specifically, the widely-used QUIC protocol [40] needed encryption to hide the nonce, and accordingly uses AE2 which (recall) is a special case of AE3/AE5. The better answer, however, is that judging what is of interest based solely on existing applications is limited and backward-looking. Prior work [8] has given compelling motivations to move from AE1 to AE3/AE5. In preparation for such a move, it behoves us as researchers to

Work	Transform	Tag Based?	AE Type	Preserves	Expansion
BH22 [4]	HtE∘UtC	No	AE1	UNAE	256
	HtE∘RtC	No	AE1	UNAE, MRAE	256
CR22 [14]	CTX	Yes	AE1	UNAE	128
BH24 [5]	CTY	Yes	AE5	UNAE	128
MBHLR25 [33]	XtH	Yes	AE5	UNAE	128
CK25 [15]	HtAE	No	AE1	UNAE	256
	AEaH	No	AE1	UNAE	256
	AEtH	No	AE1	UNAE, MRAE	256
Ours	**UCX**	**No**	**AE5**	**UNAE, MRAE**	128

Table 1: **Committing transforms**. The 3rd column indicates whether or not the starting scheme SE is restricted to be tag-based (universality requires "No"). The 4th column indicates the genre of AE for which the transform works (universality requires AE5). The 5th column indicates which of UNAE, MRAE is preserved (universality requires both). The last column shows commitment ciphertext overhead for $s = 128$ bits of CMT security assuming a 128-bit tag length for tag-based schemes and a setting of the parameter t to 128 for **UCX**.

---

have provided ways to add CMT security for AE3/AE5. As evidence of this movement, the XtH transform [33] already adds CMT to AE5 schemes. (And the OCH scheme they build [33] is both AE5 and CMT.) However, XtH only works for tag-based schemes, so is not universal.

The transform preserves both UNAE and MRAE. UNAE (Unique Nonce AE) refers to the basic form of AE where security requires that encryption never reuse a nonce. MRAE (Misuse Resistant AE) lifts this restriction to give a stronger security guarantee. Note that, while the first definitions were for AE1 [37, 39], we, having already required that we work in the AE5 framework, are referring to UNAE and MRAE definitions in this broader framework. Both UNAE and MRAE schemes exist and are of interest, and we want to be able to add CMT to either. Accordingly, the third requirement for universality of the transform **CT** is that it preserves both UNAE and MRAE: If SE is UNAE then so is $\overline{\mathsf{SE}}$, and likewise for MRAE.

The transform is efficient and flexible. CMT being subject to offline attacks, $s = 64$ bits of CMT security for $\overline{\mathsf{SE}}$ is not always considered enough. The recommendation in contemporary work is $s = 128$. For a universal transform, however, we cover all bases by asking for flexibility: The transform should be able to add any given number s of bits of CMT security. Now, while providing universality, we want of course to also minimize cost overhead in both computation and bandwidth. With regard to the former, we want computational overhead to be independent of message length. For bandwidth, we define the *committing ciphertext overhead* $co(s)$ as the length of the ciphertext in the transformed scheme minus the length of the ciphertext in the base scheme. This will be a constant depending only on s. Our target will be $co(s) \leq s$. For tag based schemes, the SC transform [5] can be further applied to reduce this, but nothing like this is known for non-tag-based schemes. (We don't have a proof that it is impossible, but we do not expect it.)

Prior transforms are not universal. We now assess prior transforms with regard to providing universality. A summary is in Table 1.

CTX [14] requires the starting scheme SE to be a tag-based AE1 scheme, and is shown to preserve only UNAE. Thus it meets none of the first three conditions for universality. CTY [5] and

XtH [33], although shown to work when SE is AE5, are only shown to preserve UNAE and also require that SE is tag based, so do not meet two of the first three universality conditions.

The HtAE and AEaH transforms of CK25 [15] do not assume SE is tag based, but they work for UNAE AE1 schemes and thus meet only one of the first three conditions for universality. Their AEtH transform [15] works for MRAE, but still only for AE1. Additionally, all these transforms have a high committing ciphertext overhead, adding 256 bits to provide 128 bits of CMT security. (CTX, CTY, XtH and our **UCX** transform only add 128 bits of overhead.)

The PACT transform of BBD24 [13] does not assume SE is tag based, and it preserves MRAE. But it only works for AE1 and has a further limitation. It requires that SE satisfy an additional property they define, called ciphertext-collision resistance (CIPH-COLL) [13]. This property is new and non-standard. Indeed the CIPH-COLL-security of existing AE schemes has not been considered, let alone proved. BBD24 [13] do show that Ascon [16] and Deoxys-II [29] are CIPH-COLL, but beyond that they only conjecture it, saying that "any well-designed AE scheme is CIPH-COLL" [13, Section 4.4]. However, in this paper, we refute this conjecture. In Appendix A, we give a simple attack breaking CIPH-COLL security of the standardized GCM [18] AE scheme. In particular this means PACT may not even work on SE = GCM. Similar CIPH-COLL attacks are possible on other AE schemes. Thus we cannot regard PACT as a viable candidate for a universal transform. comPACT [13] is a more efficient version of PACT for UNAE AE1 schemes, but continues to assume CIPH-COLL and is thus subject to the same limitation. Due to these limitations, we have for simplicity omitted PACT and comPACT from Table 1.

BH22 [4] give transforms UtC, RtC and HtE. The composition HtE∘UtC adds CMT security to UNAE AE1 schemes without assuming they are tag based, meeting only one of the first three conditions for universality. HtE∘RtC works for MRAE, but still only for AE1. However, these transforms also have an additional limitation. Namely, to add s bits of CMT security, they require that the key-length of SE be at least $2s$, which is 256 in the case $s = 128$ of most interest. This excludes important choices for SE, including the AES-128 version of GCM.

The function of the SC transform of [5] is ciphertext shortening, not adding CMT security. It takes an AE scheme SE that is already CMT-secure, and preserves this while reducing ciphertext size, assuming additonally SE is tag based. Accordingly we do not show SC in Table 1.

Our **UCX** transform. The final row of Table 1 shows our transform **UCX**. It does not assume the starting scheme SE is tag based, works for AE5 rather than just AE1, and preserves both UNAE and MRAE. Moreover, to add $s = 128$ bits of CMT security, it suffers commitment ciphertext overhead which is either the same as, or lower than, that of the other transforms.

While **UCX** does not require that SE is tag based, it has the nice property (shown in Proposition 5.1) that if SE happens to be tag based then the constructed scheme $\overline{\mathsf{SE}}$ is also tag based. This permits us, in the UNAE case, to further apply the above-mentioned SC transform of [5] to reduce commitment ciphertext overhead to close to zero.

Tweakable committing concealers and **HER**. As a tool in constructing our **UCX** transform, we employ a new primitive that we introduce, define and build. We call it a Tweakable Committing Concealer (TCC). A TCC F takes a key, tweak and input to deterministically return an output. The input is expected to be shorter than the output, and in place of invertibility we require a partial invertibility condition. We ask for two security conditions. The first is collision resistance (CR) and the second, PRI (Pseudo Random Injection) security [39], is a generalization of strong-PRP security to partially-invertible functions. We then give a construction of a TCC that we call **HER**. It uses a keyed hash function H and a block cipher E. Theorem 4.3 establishes CR security of **HER** with good bounds. It assumes H is itself CR, and E is an ideal cipher. Theorem 4.4 establishes PRI security of **HER** with good bounds. It assumes H is a PRF and

E is a strong-PRP. In particular, it makes only standard-model assumptions, no idealized ones. **HER** is additionally efficient. A typical instantiation for 128-bit security would use Rijndael-256 for E. (If this raises the concern of possible use of a non-standardized algorithm, note that NIST proposed in December 2024 to standardize Rijndael-256 [21] and we would expect that it is soon a standard.) For H, we can use truncated HMAC-SHA512. (The latter is already a standard [20, 19].) See Section 4 for formal definitions, a full description of **HER**, theorems and proofs.

How do TCCs relate to other primitives? The starting point is Committing Concealers (CCs) as introduced by Hoang and Menda [28]. These however, have no tweaks. Also, while a CC is a block cipher, a TCC, even disregarding the tweak, is not, since the inputs can be shorter than the outputs. The security properties of a CC are hiding and binding as defined in [28]. Binding is like CR but hiding, which is parameterized by an encoding function, is different from PRI security. One can also ask how TCCs relate to Tweakable Block Ciphers (TBCs) [32]. Syntactically, a TBC is the special case of a TCC in which the input and output length are the same. However, no TBC can achieve our CR security condition (see Section 4 for an attack), meaning no TBC is, by itself, a secure TCC.

THE **UCX** TRANSFORM. Our **UCX** transform uses, as auxiliary primitives, a tweakable committing concealer (TCC) F and a key-derivation function KD. Given these, as well as the starting AE scheme SE and an integer parameter t, it returns a cAE scheme $\overline{\mathsf{SE}} = \mathbf{UCX}[\mathsf{SE}, \mathsf{F}, \mathsf{KD}, t]$ as shown in Figure 10. The key-derivation function is used to derive, from the main key K, an encryption sub-key EK to be used with SE, and a commitment sub-key CK to be used with F. The requirements we formalize for KD are PR (asking that both sub-keys are pseudorandom) and CMTK (asking that CK commits to K). The TCC F is instantiated via **HER**. The key-derivation function KD is easily instantiated, for example via HMAC-SHA512. Recall that here SE is an AE5 scheme, so the nonce $N = NP\|NS$ has public and secret parts, that the transform handles differently. The SE-ciphertext is split as $C'\|D$ with $|D| = t$. (We stress that, since SE is not assumed to be tag based, D is not necessarily a tag.) The prefix C' is returned as the prefix of the $\overline{\mathsf{SE}}$-ciphertext. The suffix D is "stolen" as the input for the TCC F, which, with the tweak being the public nonce and associated data, returns the suffix of the $\overline{\mathsf{SE}}$-ciphertext. Theorem 5.2 shows that $\overline{\mathsf{SE}}$ is CMT secure, assuming only the collision-resistance of F (provided by **HER**, in the ideal cipher model, as per Theorem 4.3) and the committing security of KD. Theorem 5.3 shows that **UCX** preserves both UNAE and MRAE security of SE, meaning, for both $\mathrm{X} \in \{\mathrm{UNAE}, \mathrm{MRAE}\}$, if SE is X-secure than so is $\overline{\mathsf{SE}}$. This is shown in a unified way. The assumptions are the PRI security of the TCC F (provided by **HER** as per Theorem 4.4) and the PR-security of KD.

FAQ. Here we address some questions and objections from reviews across two submissions of this paper.

1. We already have specialized transforms that work in special cases, so why would we want a universal one? As above, on the theoretical side, it provides a clean and general solution. On the practical side, it can reduce misuse and error. If a cryptographic software library provides many specialized transforms, an implementor has to make a choice. They could make an incorrect one, or one that is obviated by a later change to the base scheme. If there is just one transform, they can't go wrong.

2. The motivation for targeting AE3/AE5 is weak. What applications benefit? The widely-deployed QUIC protocol [40] uses a scheme that is AE2. (A special case of AE3/AE5.) AE5 was also a requirement for the CAESAR competition. (Even though schemes ended up not meeting it.) But, more broadly and importantly, current applications should not be the only motivation. Prior work [8] has pointed out salient reasons for which AE1 is flawed and AE3/AE5 are superior.

In particular, contrary to common understanding, or advertisement, AE1 fails to provide even message security for arbitrary (non-repeating) nonces [8]. The continued use of AE1 in the face of this represents inertia and legacy rather than informed choice. We believe that AE3/AE5 is the right target moving forward. The quest for AE5 committing schemes has already started with the XtH transform and the OCH [33] scheme, but the former is not universal, a gap we fill.

3. Yes, **UCX** works for non-tag based schemes, but are there such schemes? As discussed above there are important non-tag-based schemes including SIV-based ones like AES-GCM-SIV [26] and EtE-based ones like AEZ [27].

2 Preliminaries

NOTATION AND TERMINOLOGY. Let ε denote the empty string. For a string $x \in \{0,1\}^*$ we let $|x|$ denote its length, and let $x[i : j]$ denote the bits in positions i through j (inclusive), for $1 \le i \le j \le |x|$. For an integer $n \ge 1$, we let $\{0,1\}^{\le n} = \{\, x \in \{0,1\}^* : |x| \le n \,\}$ be the set of strings of length at most n. We use $\perp$ as a special symbol to denote rejection, and it is assumed to be outside $\{0,1\}^*$.

By $\mathrm{FUNC}(\mathsf{Dom}, \mathsf{Rng})$ we denote the set of all functions $f : \mathsf{Dom} \to \mathsf{Rng}$. If $f \in \mathrm{FUNC}(\mathsf{Dom}, \mathsf{Rng})$ is a function then its image is defined as $\mathsf{Img}(f) = \{\, f(x) : x \in \mathsf{Dom} \,\} \subseteq \mathsf{Rng}$. By $\mathrm{PERM}(n)$ we denote the set of all permutations $\pi : \{0,1\}^n \to \{0,1\}^n$. If $I : \{0,1\}^t \to \{0,1\}^n$ is an injective function then we define its inverse $I^{-1} : \{0,1\}^n \to \{0,1\}^t \cup \{\perp\}$ as follows: Let $I^{-1}(I(x)) = x$ for all $x \in \{0,1\}^t$ and let $I(y) = \perp$ for all $y \in \{0,1\}^n \setminus \mathsf{Img}(I)$. Note I^{-1} can return $\perp$. We let $\mathrm{INJ}(t, n)$ denote the set of all injections (injective functions) from $\{0,1\}^t$ to $\{0,1\}^n$.

If X is a finite set, we let $x \leftarrow\!\$\, X$ denote picking an element of X uniformly at random and assigning it to x. If A is a randomized algorithm, we let $y \leftarrow\!\$\, A(x_1, \ldots)$ denote running A on inputs $x_1, \ldots$ and assigning the output to y. If A is deterministic, we may use $\leftarrow$ in place of $\leftarrow\!\$$. If S is a finite set, then $|S|$ denotes it size. We say that a set S is *full* if it is either $\{0,1\}^*$ or $\{0,1\}^{\le n}$ for some n. (This will be a requirement for message, associated-data, and nonce spaces.)

For two adversaries that are given the same set of oracles, we say that they have the same *query statistics* if they have the same number of queries, the same number of queries per user, and the same total length of queries, for each oracle that they are given.

GAMES. We use the code-based game-playing framework of [10] with some modifications. A game $\mathrm{G}(\mathcal{A})$ runs adversary $\mathcal{A}$ with oracles that the game specifies as procedures, and then returns an output. By $\Pr[\mathrm{G}(\mathcal{A}) \Rightarrow y]$ we denote the probability that the execution of $\mathrm{G}(\mathcal{A})$ results in the game output being y, and write just $\Pr[\mathrm{G}(\mathcal{A})]$ for $\Pr[\mathrm{G}(\mathcal{A}) \Rightarrow \mathsf{true}]$. In game pseudocode, integer variables, set variables, boolean variables and string variables are assumed initialized, respectively, to 0, the empty set $\emptyset$, the boolean false and $\perp$. Adversaries in games are always assumed to be domain-respecting, meaning if a query they provide is expected to fall in some scheme-associated set, then it does. For the following, recall that games G, H are identical-until-bad if their code differs only in statements that follow the setting of flag bad to true [10].

Lemma 2.1 (Fundamental Lemma of Game Playing [10]) *Let* G, H *be identical-until-*bad *games. Then for any adversary* $\mathcal{A}$ *we have*

$$|\Pr[\mathrm{G}(\mathcal{A})] - \Pr[\mathrm{H}(\mathcal{A})]| \le \Pr[\mathrm{H}(\mathcal{A}) \textit{ sets } \mathsf{bad}\,] = \Pr[\mathrm{G}(\mathcal{A}) \textit{ sets } \mathsf{bad}\,] \,.$$

COLLISION RESISTANCE. Let $\mathsf{G} : \mathsf{Dom}_1 \times \cdots \times \mathsf{Dom}_n \to \mathsf{Rng}$ be a function. A *collision* for G is a pair (X_1, X_2) such that $X_1, X_2 \in \mathsf{Dom}_1 \times \cdots \times \mathsf{Dom}_n$ and $X_1 \ne X_2$ but $\mathsf{G}(X_1) = \mathsf{G}(X_2)$.

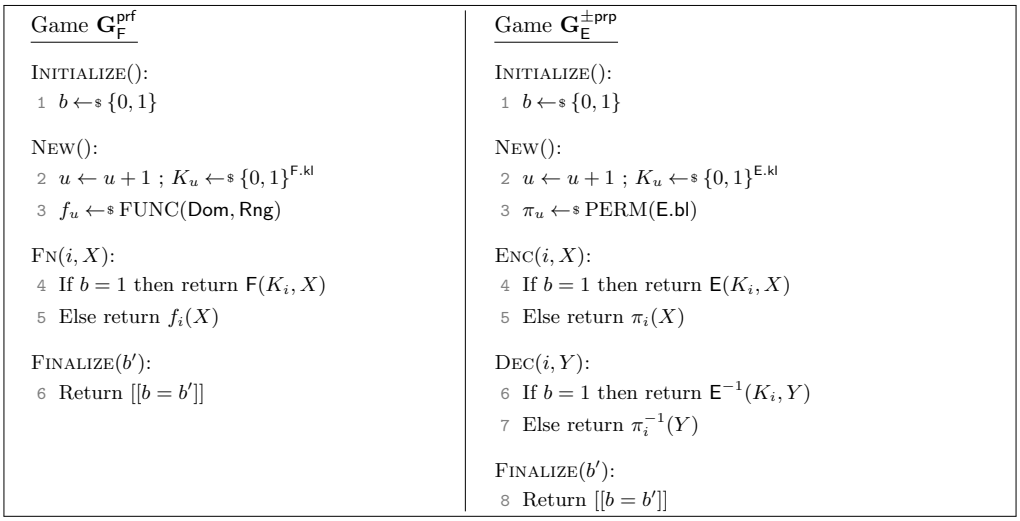


Figure 1: Left: Game defining (multi-user) PRF security of function family F. Right: Game defining (multi-user) strong-PRP security of blockcipher E.

For an adversary $\mathcal{A}$, define its advantage in breaking the collision resistance of G as $\mathbf{Adv}^{\mathrm{cr}}_{\mathsf{G}}(\mathcal{A}) = \Pr[(X_1,X_2) \text{ is a collision for } \mathsf{G}]$, where the probability is over $(X_1,X_2) \leftarrow\$ \mathcal{A}$.

PRFs and PRPs. For a function $\mathsf{F} : \{0,1\}^{\mathsf{F.kl}} \times \mathsf{Dom} \to \mathsf{Rng}$ and an adversary $\mathcal{A}$, we define the advantage of $\mathcal{A}$ in breaking the (multi-user) PRF security of F as $\mathbf{Adv}^{\mathrm{prf}}_{\mathsf{F}}(\mathcal{A}) = 2\Pr[\mathbf{G}^{\mathsf{prf}}_{\mathsf{F}}(\mathcal{A})] - 1$, where game $\mathbf{G}^{\mathsf{prf}}_{\mathsf{F}}$ is shown in Fig. 1.

For a blockcipher $\mathsf{E} : \{0,1\}^{\mathsf{E.kl}} \times \{0,1\}^{\mathsf{E.bl}} \to \{0,1\}^{\mathsf{E.bl}}$ and an adversary $\mathcal{A}$, we define the advantage of $\mathcal{A}$ in breaking the (multi-user) strong-PRP security of E as $\mathbf{Adv}^{\pm\mathrm{prp}}_{\mathsf{E}}(\mathcal{A}) = 2\Pr[\mathbf{G}^{\pm\mathsf{prp}}_{\mathsf{E}}(\mathcal{A})] - 1$, where game $\mathbf{G}^{\pm\mathsf{prp}}_{\mathsf{E}}$ is shown in Fig. 1.

Multi-user RI/RF switching lemma. The PRP/PRF switching lemma [6, 10] considers an adversary $\mathcal{A}$ making q queries to an oracle that is either a random permutation or a random function on n bits, and bounds its distinguishing advantage by $q(q-1)/2^{n+1}$. We need, and give here, a variant where the oracle is either a random injection (RI) from t to n bits or a random function (RF) from t to n bits. Additionally, we are interested in the multi-user setting. Using the standard hybrid argument to derive multi-user security from single-user security gives a weak bound. A switching lemma with a better multi-user bound, for the conventional PRP/PRF case, is given in [4]. We extend this to get a multi-user RI/RF switching lemma with a good bound. It considers the games of Fig. 2. The lemma is then as follows.

Lemma 2.2 (Multi-User RI/RF Switching Lemma) *Let $1 \le t \le n$. Let $\mathcal{B}$ be an adversary that makes q Fn queries with at most B of these per user. Then*

$$\left| \Pr[\mathbf{G}^{\mathrm{ri}}_{n}(\mathcal{B})] - \Pr[\mathbf{G}^{\mathrm{rf}}_{n}(\mathcal{B})] \right| \le \frac{qB}{2^n} \ .$$

Proof of Lemma 2.2: The proof is analogous to that of the multi-user PRP/PRF switching lemma of [4]. Briefly, the view of $\mathcal{B}$ is the same unless a bad event happens, the latter being that for some i there are collisions in queries to f_i in game $\mathbf{G}^{\mathrm{rf}}_{t,n}$. Since there are at most B queries per user, each of the q queries has chance at most $B/2^n$ of creating such a collision. Then we apply the union bound across the q queries. ▌

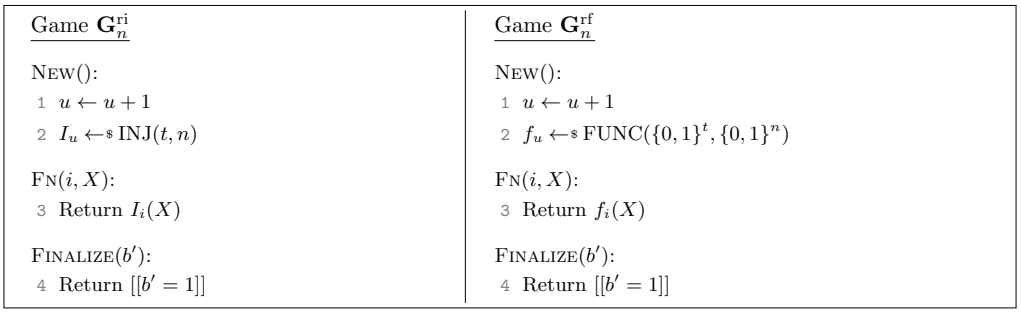


Figure 2: Games for multi-user PRP/PRF switching lemma.

3 AE Framework

Many forms of nonce-based authenticated encryption with associated data have been considered in the literature. To maximize the applicability (universality) of our committing transform, we use the general AE5 [36] definition. This is a somewhat simpler variant of the AE3 [5] definition. It in particular includes AE1 [37] and AE2 [8] as special cases, but also covers broader applications than either of these two.

Syntax. A symmetric encryption scheme SE specifies a deterministic encryption algorithm $\mathsf{SE.Enc} : \{0,1\}^{\mathsf{SE.kl}} \times \{0,1\}^{\mathsf{SE.pnl}} \times \{0,1\}^{\mathsf{SE.snl}} \times \mathsf{SE.AS} \times \mathsf{SE.MS} \to \{0,1\}^*$ and a deterministic decryption algorithm $\mathsf{SE.Dec} : \{0,1\}^{\mathsf{SE.kl}} \times \{0,1\}^{\mathsf{SE.pnl}} \times \mathsf{SE.AS} \times \{0,1\}^* \to (\mathsf{SE.MS} \times \{0,1\}^{\mathsf{SE.snl}}) \cup \{\perp\}$. Here $\mathsf{kl}, \mathsf{pnl}, \mathsf{snl}$ are the key, public-nonce and secret-nonce lengths, respectively. The associated-data space $\mathsf{SE.AS}$ and message space $\mathsf{SE.MS}$ are required to be full sets as defined above. We define the nonce space as $\{0,1\}^{\mathsf{SE.nl}} = \{0,1\}^{\mathsf{SE.pnl}} \times \{0,1\}^{\mathsf{SE.snl}}$, meaning a nonce is a pair $N = NP \| NS$ consisting of a public nonce and a secret nonce. We require that there is a *ciphertext-length function* $\mathsf{SE.cl} : \mathbb{N} \to \mathbb{N}$ such that $|\mathsf{SE.Enc}(K, NP, NS, A, M)| = \mathsf{SE.cl}(|M|)$ for all $(K, NP, NS, A, M) \in \{0,1\}^{\mathsf{SE.kl}} \times \{0,1\}^{\mathsf{SE.pnl}} \times \{0,1\}^{\mathsf{SE.snl}} \times \mathsf{SE.AS} \times \mathsf{SE.MS}$. The *ciphertext-expansion function* $\mathsf{SE.ce} : \mathbb{N} \to \mathbb{N}$ is then defined by $\mathsf{SE.ce}(m) = \mathsf{SE.cl}(m) - m - \mathsf{SE.snl}$ for all $m \in \mathbb{N}$. (Since decryption recovers the secret nonce, it should be considered like the message when defining ciphertext expansion. Note ciphertext expansion is different from what we called commitment ciphertext overhead in Section 1. The latter is defined for transforms, not schemes.) The correctness requirement is that if $C \leftarrow \mathsf{SE.Enc}(K, NP, NS, A, M)$ then $\mathsf{SE.Dec}(K, NP, A, C)$ returns (M, NS) for all $(K, NP, NS, A, M) \in \{0,1\}^{\mathsf{SE.kl}} \times \{0,1\}^{\mathsf{SE.pnl}} \times \{0,1\}^{\mathsf{SE.snl}} \times \mathsf{SE.AS} \times \mathsf{SE.MS}$.

To explain, the sender as usual sends the ciphertext $C \leftarrow \mathsf{SE.Enc}(K, NP, NS, A, M)$ to the receiver. However, what information the receiver gets about the nonce varies across definitions. In AE1 (also called AEAD [37]), the entire nonce is sent in the clear and is an explicit input to $\mathsf{SE.Dec}$. This is captured as a special case of the above syntax, namely there is no secret nonce; formally, $\mathsf{SE.snl} = 0$. The AE1 definition however fails to provide message privacy for some choices of nonces, for example when they are hashes of the messages [8], and also fails to hide the nonce, which in some applications can be sensitive, for example a device id. This lead [8] to introduce AE2 (also called nonce-hiding AE), where the decryptor gets no information about the nonce, captured in the above syntax as the special case in which $\mathsf{SE.pnl} = 0$. However, in applications, the nonce can consist of a sensitive part, meaning one whose privacy we want to protect, like a device-id, and a non-sensitive part, meaning one whose privacy is not a concern, like a random number. Neither AE1 nor AE2 support this, but the above syntax does, by setting NP to the non-sensitive part of the nonce and NS to the sensitive part. In this framework, the public nonce is sent in the clear with the ciphertext and is an input to the decryption algorithm, while the secret nonce is recovered

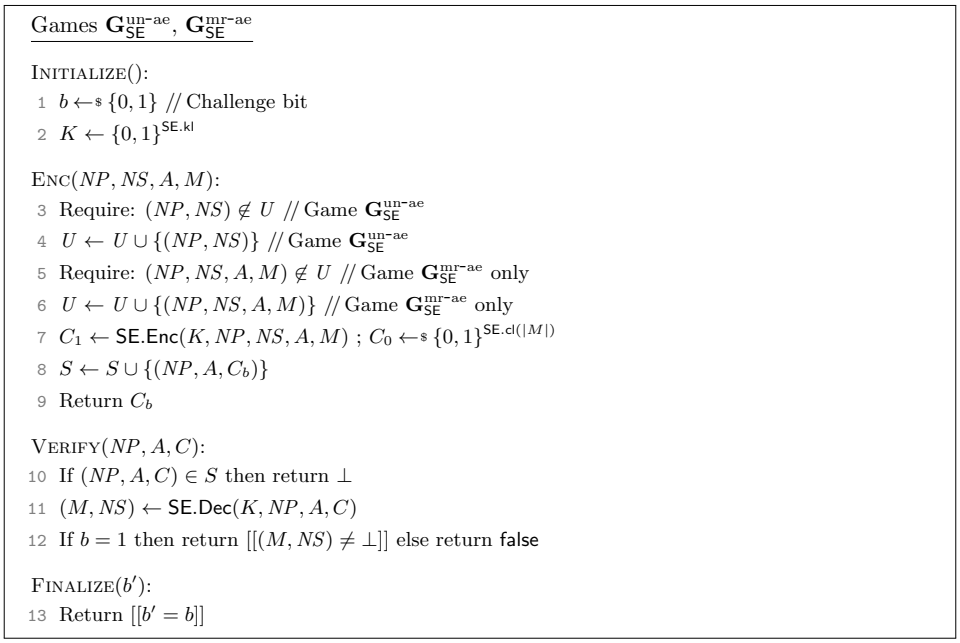


Figure 3: Games defining the UNAE and MRAE security of a symmetric encryption scheme SE.

in decryption and remains hidden.

We require tidiness as defined in [5], extending the AE1 definition of [39]. Namely, for all $(K, NP, A, C) \in \{0,1\}^{\mathsf{SE.kl}} \times \{0,1\}^{\mathsf{SE.pnl}} \times \mathsf{SE.AS} \times \{0,1\}^*$ the following is true: If $(M, NS) \leftarrow \mathsf{SE.Dec}(K, NP, A, C)$ and $(M, NS) \neq \perp$, then it must be that $\mathsf{SE.Enc}(K, NP, NS, A, M) = C$.

We say that symmetric encryption scheme SE is *tag based* if its ciphertext $C = C^* \| T$ consists of a ciphertext core C^* and a SE.tl-bit tag T that decryption recomputes from C^* and verifies. Formally, SE is tag based if it specifies an additional deterministic tagging algorithm $\mathsf{SE.Tag} : \{0,1\}^{\mathsf{SE.kl}} \times \{0,1\}^{\mathsf{SE.pnl}} \times \mathsf{SE.AS} \times \{0,1\}^* \to \mathsf{SE.MS} \times \{0,1\}^{\mathsf{SE.snl}} \times \{0,1\}^{\mathsf{SE.tl}}$ in terms of which the decryption algorithm is specified as follows:

$\underline{\mathsf{SE.Dec}(K, NP, A, C)}$
If $|C| < \mathsf{SE.tl}$ then return $\perp$
$C^* \| T \leftarrow C$ // $|T| = \mathsf{SE.tl}$
$(M, NS, T^*) \leftarrow \mathsf{SE.Tag}(K, NP, A, C^*)$
If $(T^* = T)$ then return (M, NS) else return $\perp$

We refer to SE.tl as the tag length. Encrypt-then-MAC schemes [7, 35] like GCM and CCM are tag-based, but as we have discussed, there are important schemes that are not tag based, and we want our committing transform to work on those as well.

AE security. Let SE be a symmetric encryption scheme as per the syntax above. We define AE security of SE following the formalization of AE3 [5]. We define both basic security, called unique-nonce or UNAE, where encryption nonces may not repeat, and advanced security, called misuse resistance [39], where this restriction is lifted. Our transform will work in both cases.

Fig. 3 specifies two games. Lines 3,4 are included only in game $\mathbf{G}^{\text{un-ae}}_{\mathsf{SE}}$, which defines unique-nonce security. This game requires that a nonce —this means the pair $NP \| NS$— cannot be repeated in an Enc query. Lines 5,6 are included only in game $\mathbf{G}^{\text{mr-ae}}_{\mathsf{SE}}$, which defines misuse-resistance security. This game requires only that a full query to Enc not repeat. We stress that

Game $\mathbf{G}^{\mathrm{cmt}}_{\mathsf{SE}}$

FINALIZE$((K_1, NP_1, NS_1, A_1, M_1), (K_2, NP_2, NS_2, A_2, M_2))$:
1 $C_1 \leftarrow \mathsf{SE.Enc}(K_1, NP_1, NS_1, A_1, M_1)$; $C_2 \leftarrow \mathsf{SE.Enc}(K_2, NP_2, NS_2, A_2, M_2)$
2 If $(C_1 \neq C_2)$ then return false
3 Return $((K_1, NP_1, NS_1, A_1, M_1) \neq (K_2, NP_2, NS_2, A_2, M_2))$

Figure 4: Game defining committing security of a symmetric encryption scheme SE.

there are no such restrictions on verification queries. We ask of course that queries are restricted to their domains, for example that $(NP, NS, A, M) \in \{0,1\}^{\mathsf{SE.pnl}} \times \{0,1\}^{\mathsf{SE.snl}} \times \mathsf{SE.AS} \times \mathsf{SE.MS}$ in an ENC(NP, NS, A, M) query. For xx $\in \{\mathrm{un}, \mathrm{mr}\}$ we define the xx-ae advantage of an adversary $\mathcal{A}$ as $\mathbf{Adv}^{\text{xx-ae}}_{\mathsf{SE}}(A) = 2 \cdot \Pr[\mathbf{G}^{\text{xx-ae}}_{\mathsf{SE}}(A)] - 1$.

COMMITTING SECURITY. We ask for strong committing security where all inputs to the encryption algorithm are committed to by a ciphertext. This was first defined as CMT-4 in [4] and CAE-XX in [14]. It is called context-committing security in [34]. These definitions were for AE1. We need a definition for our AE5 syntax. This was given in [5]. We will simply call it committing security, abbreviated CMT. For an adversary $\mathcal{A}$, we define its advantage in breaking the committing security of symmetric encryption scheme SE as $\mathbf{Adv}^{\mathrm{cmt}}_{\mathsf{SE}}(\mathcal{A}) = \Pr[\mathbf{G}^{\mathrm{cmt}}_{\mathsf{SE}}(\mathcal{A})]$, where game $\mathbf{G}^{\mathrm{cmt}}_{\mathsf{SE}}(\mathcal{A})$ is defined in Fig. 4. Informally, committing security asks that the encryption algorithm be collision resistant.

An alternative, decryption based version of committing security is defined in [5], but they show it is equivalent to the encryption based version we have defined for tidy schemes, so for simplicity we use the encryption based version.

4 Tweakable committing concealers and HER

Committing concealers (CCs) were introduced in [28]. A CC is a block cipher satisfying a weak form of ±prp security and a binding property. Here we define and construct a primitive that we call a tweakable committing concealer (TCC). It extends CCs in several ways. First, it is not a block cipher, but rather a function family with partial invertibility. Second, as the name indicates, there is a tweak. Third, the security properties are strengthened. After we have defined TCCs, we will discuss their relation to CCs, as well as to other related primitives like tweakable block ciphers (TBCs) [32] and CR invertible-PRFs [5], in more detail. Then we give, and prove secure, our **HER** TCC construction.

SYNTAX. A tweakable committing concealer (TCC) F describes a deterministic evaluation algorithm $\mathsf{F.Ev} : \{0,1\}^{\mathsf{F.kl}} \times \mathsf{F.T} \times \{0,1\}^{\mathsf{F.il}} \to \{0,1\}^{\mathsf{F.ol}}$ where $\mathsf{F.kl}$ is the key length, $\mathsf{F.T}$ is the tweak space, $\mathsf{F.il}$ is the input length and $\mathsf{F.ol}$ is the output length. The invertibility is that there is a deterministic inverse algorithm $\mathsf{F.Inv} : \{0,1\}^{\mathsf{F.kl}} \times \mathsf{F.T} \times \{0,1\}^* \to \{0,1\}^{\mathsf{F.il}} \cup \{\perp\}$ such that $\mathsf{F.Inv}(CK, T, \mathsf{F.Ev}(CK, T, D)) = D$ for all $(CK, T, D) \in \{0,1\}^{\mathsf{F.kl}} \times \mathsf{F.T} \times \{0,1\}^{\mathsf{F.il}}$ and also $\mathsf{F.Inv}(CK, T, Y) = \perp$ if there is no $D \in \{0,1\}^{\mathsf{F.il}}$ such that $Y = \mathsf{F}(CK, T, D)$. A tweakable blockcipher (TBC) [32] is the special case in which $\mathsf{F.il} = \mathsf{F.ol}$, and, in that case, $\mathsf{F.Inv}$ never returns $\perp$. We will however construct and use TCCs that are not TBCs because the latter do not meet our desired security conditions.

SECURITY. We will need TCCs satisfying two security conditions that we define, namely collision resistance (cr) and PRI (pseudorandom injection) security.

In Section 2, we defined the collision resistance (cr) advantage $\mathbf{Adv}^{\mathrm{cr}}_{\mathsf{G}}(\mathcal{A})$ of an adversary $\mathcal{A}$ attacking an arbitrary function $\mathsf{G} : \mathsf{Dom}_1 \times \cdots \times \mathsf{Dom}_n \to \mathsf{Rng}$. Collision resistance of a TCC F is

Game $\mathbf{G}^{\text{pri}}_{\mathsf{F}}$

INITIALIZE:
1 $CK \xleftarrow{\$} \{0,1\}^{\mathsf{F.kl}}$; $b \xleftarrow{\$} \{0,1\}$
2 For $T \in \mathsf{F.T}$ do $I_T \xleftarrow{\$} \text{INJ}(\mathsf{F.il}, \mathsf{F.ol})$

FN(T, D):
3 $Y_1 \leftarrow \mathsf{F.Ev}(CK, T, D)$; $Y_0 \leftarrow I_T(D)$
4 Return Y_b

INV(T, Y):
5 $D_1 \leftarrow \mathsf{F.Inv}(CK, T, Y)$; $D_0 \leftarrow I_T^{-1}(Y)$
6 Return D_b

FINALIZE(b'):
7 Return $[[b = b']]$

$\mathbf{HER}[\mathsf{H}, \mathsf{E}, t].\mathsf{Ev}(K, T, D)$:
1 $L \leftarrow \mathsf{H}(K, T)$
2 $Y \leftarrow \mathsf{E}(L, D \| 0^{n-t})$
3 Return Y

$\mathbf{HER}[\mathsf{H}, \mathsf{E}, t].\mathsf{Inv}(K, T, Y)$:
4 $L \leftarrow \mathsf{H}(K, T)$
5 $C \leftarrow \mathsf{E}^{-1}(L, Y)$
6 $X \| P \leftarrow C \;//\; |P| = n - t$
7 If $(P = 0^{n-t})$ then return X else return $\perp$

Figure 5: **Left:** Game defining PRI security of F. **Right:** Our $\mathbf{HER}[\mathsf{H}, \mathsf{E}, t]$ TCC.

then defined simply as that of $\mathsf{F.Ev}$, and for convenience we let $\mathbf{Adv}^{\text{cr}}_{\mathsf{F}}(\mathcal{A}) = \mathbf{Adv}^{\text{cr}}_{\mathsf{F.Ev}}(\mathcal{A})$.

We define PRI (Pseudo Random Injection) security of a TCC F using the game $\mathbf{G}^{\text{pri}}_{\mathsf{F}}$ shown in Figure 5 [39]. The advantage of adversary $\mathcal{A}$ is defined as $\mathbf{Adv}^{\text{pri}}_{\mathsf{F}}(\mathcal{A}) = 2 \cdot \Pr[\mathbf{G}^{\text{pri}}_{\mathsf{F}}(\mathcal{A})] - 1$. To explain, for each tweak T, the game picks a random injection $I_T : \{0,1\}^{\mathsf{F.il}} \to \{0,1\}^{\mathsf{F.ol}}$. It then asks that, for a hidden key CK, oracles for $\mathsf{F.Ev}(CK, T, \cdot), \mathsf{F.Inv}(CK, T, \cdot)$ be indistinguishable from oracles for I_T, I_T^{-1}, even when the adversary can choose T. When $\mathsf{F.il} = \mathsf{F.ol}$, this coincides with tweakable strong PRP security, but the interesting and relevant case for us is that $\mathsf{F.il} < \mathsf{F.ol}$, in particular $s = \mathsf{F.ol} - \mathsf{F.il}$ is large. Now recall that our definition of I_T being an injection means that I_T^{-1} returns $\perp$ on inputs outside the image of I_T, and when s is large there are lots of such inputs. So we expect that if the adversary makes a non-trivial $\text{INV}(T, Y)$ query (meaning no prior $\text{FN}(T, D)$ query had returned Y) then it will get back $\perp$. So the definition ends up asking that, for hidden CK, it is hard to find T, Y such that $\mathsf{F.Inv}(CK, T, Y) \neq \perp$, except trivially.

RELATION TO OTHER PRIMITIVES. We discuss how TCCs are different from related primitives. First, since we already noted that any tweakable blockcipher (TBC) is, syntactically, a TCC, a first question may be, why can we not just use a TBC as the TCC? The answer is that a TBC F is *not* CR, as shown by the following, simple attack. The adversary can pick an arbitrary CK, D_1 and distinct T_1, T_2 and let $D_2 \leftarrow \mathsf{F.Inv}(CK, T_2, \mathsf{F.Ev}(CK, T_1, D_1))$ and $(CK, T_1, D_1), (CK, T_2, D_2)$ will be a collision.

A committing concealer (CC) as defined in [28] is a block cipher. It has no tweak. Also as defined in [28], committing security is parameterized by an encoding function, which is different from our CR requirement.

Invertible PRFs as defined in [5] do not have a tweak. Even if the tweak is dropped, our PRI notion is different from their IPF one.

If we think of the tweak as associated data then a TCC could be seen as a committing, robust AE scheme [27, 28] that has no nonce and fixed-length messages, with only one allowed tag length.

THE **HER** CONSTRUCTION. We give a construction of a CR and PRI secure TCC that we call **H**ash then **E**ncipher with **R**edundancy (**HER**). Let $\mathsf{H} : \{0,1\}^k \times \{0,1\}^* \to \{0,1\}^\ell$ be a function with output length ℓ, and let $\mathsf{E} : \{0,1\}^\ell \times \{0,1\}^n \to \{0,1\}^n$ be a blockcipher with key length ℓ and block length n. The construction is formally a transform **HER** that takes H, E and an integer $t < n$ (the input length) to return a TCC $\mathsf{F} = \mathbf{HER}[\mathsf{H}, \mathsf{E}, t]$. The evaluation and inverse algorithms of F are shown in Figure 5, and the associated parameters and sets are $\mathsf{F.kl} = k$ and $\mathsf{F.T} = \{0,1\}^*$

and $\mathsf{F.il} = t$ and $\mathsf{F.ol} = n$. The inclusion of redundancy at line 2 follows the encode-then-encipher paradigm [9].

For 128-bit security we need $n = \ell = 256$. A suggested instantiation is to use Rijndael-256 for E, and, for H use HMAC-SHA512 with output truncated to 256 bits. Note that HMAC [19] and SHA512 [20] are both NIST standards, and NIST is currently in the process of standardizing Rijndael-256 [21]. In place of the latter we could also use ThreeFish [23] with empty tweak. Other instantiations are also possible.

CR Security. The following shows the CR security of TCC $\mathsf{F} = \mathbf{HER}[\mathsf{H}, \mathsf{E}, t]$ assuming H is collision resistant and E is an ideal-cipher. Towards this it is helpful to define an auxiliary **E**ncipher with **R**edundancy (**ER**) transform that takes blockcipher $\mathsf{E} : \{0,1\}^\ell \times \{0,1\}^n \to \{0,1\}^n$ and $t < n$ to return $\mathsf{G} = \mathbf{ER}[\mathsf{E}, t] : \{0,1\}^\ell \times \{0,1\}^t \to \{0,1\}^n$ defined by $\mathsf{G}(L, X) = \mathsf{E}(L, X \| 0^{n-t})$. Now notice that we can write $\mathsf{F.Ev}$ as $\mathsf{F.Ev}(K, T, D) = \mathsf{G}(\mathsf{H}(K, T), D)$. The following Lemma uses this to say that if G and H are CR then so is F. Note this Lemma does not assume E is ideal, but applies in that case too. That is, it is valid both in the standard and the ideal-cipher models.

Lemma 4.1 *Let $\mathsf{H} : \{0,1\}^k \times \{0,1\}^* \to \{0,1\}^\ell$ be a function, let $\mathsf{E} : \{0,1\}^\ell \times \{0,1\}^n \to \{0,1\}^n$ be a blockcipher and let $1 \le t < n$. Let $\mathsf{F} = \mathbf{HER}[\mathsf{H}, \mathsf{E}, t]$ and $\mathsf{G} = \mathbf{ER}[\mathsf{E}, t]$. For any adversary $\mathcal{A}_\mathsf{F}$ we can construct adversaries $\mathcal{A}_\mathsf{H}, \mathcal{A}_\mathsf{G}$ such that*

$$\mathbf{Adv}^{\mathrm{cr}}_{\mathsf{F}}(\mathcal{A}_\mathsf{F}) \le \mathbf{Adv}^{\mathrm{cr}}_{\mathsf{H}}(\mathcal{A}_\mathsf{H}) + \mathbf{Adv}^{\mathrm{cr}}_{\mathsf{G}}(\mathcal{A}_\mathsf{G}) \,. \tag{1}$$

The running times of $\mathcal{A}_\mathsf{H}$ and $\mathcal{A}_\mathsf{G}$ are similar to that of $\mathcal{A}_\mathsf{F}$. Furthermore if E is modeled as an ideal cipher to which $\mathcal{A}$ makes q queries, then $\mathcal{A}_\mathsf{G}$ also makes q queries to E.

Proof of Lemma 4.1: Adversary $\mathcal{A}_\mathsf{G}$ playing the CR game for G using $\mathcal{A}_\mathsf{F}$ does the following,

1. $((K_1, T_1, D_1), (K_2, T_2, D_2)) \xleftarrow{\$} \mathcal{A}_\mathsf{F}$
2. $L_1 \leftarrow \mathsf{H}(K_1, T_1)$; $L_2 \leftarrow \mathsf{H}(K_2, T_2)$
3. Return $((L_1, D_1), (L_2, D_2))$

The adversary $\mathcal{A}_\mathsf{H}$ playing the CR game for H using $\mathcal{A}_\mathsf{F}$ simply runs $\mathcal{A}_\mathsf{F}$ to get $((K_1, T_1, D_1), (K_2, T_2, D_2))$ and returns $((K_1, T_1), (K_2, T_2))$.

Let EV be the event that $\mathsf{F}(K_1, T_1, D_1) = \mathsf{F}(K_2, T_2, D_2)$ and $(K_1, T_1, D_1) \ne (K_2, T_2, D_2)$. Then, $\mathbf{Adv}^{\mathrm{cr}}_{\mathsf{F}}(\mathcal{A}_\mathsf{F}) = \Pr[\mathrm{EV}]$. Let EV_1 be the event that $\mathsf{H}(K_1, T_1) \ne \mathsf{H}(K_2, T_2)$. The events are defined over the outputs from $\mathcal{A}_\mathsf{F}$. We can write,

$$\mathbf{Adv}^{\mathrm{cr}}_{\mathsf{F}}(\mathcal{A}_\mathsf{F}) = \Pr[\mathrm{EV} \cap \mathrm{EV}_1] + \Pr[\mathrm{EV} \cap \overline{\mathrm{EV}_1}] \,.$$

The event $\mathrm{EV} \cap \mathrm{EV}_1$ means $L_1 \ne L_2$ and $\mathsf{G}(L_1, D_1) = \mathsf{G}(L_2, D_2)$. Therefore, $\Pr[\mathrm{EV} \cap \mathrm{EV}_1] \le \mathbf{Adv}^{\mathrm{cr}}_{\mathsf{G}}(\mathcal{A}_\mathsf{G})$.

The event $\mathrm{EV} \cap \overline{\mathrm{EV}_1}$ means that $(K_1, T_1, D_1) \ne (K_2, T_2, D_2)$ and $\mathsf{H}(K_1, T_1) = \mathsf{H}(K_2, T_2)$. Note that $(K_1, T_1, D_1) \ne (K_2, T_2, D_2)$ implies $(K_1, T_1) \ne (K_2, T_2)$ because E is a block cipher. Therefore, $\Pr[\mathrm{EV} \cap \overline{\mathrm{EV}_1}] \le \mathbf{Adv}^{\mathrm{cr}}_{\mathsf{H}}(\mathcal{A}_\mathsf{H})$ which gives us the claimed result. ▌

We are assuming H is CR, so given Lemma 4.1 it now suffices to show that $\mathsf{G} = \mathbf{ER}[\mathsf{E}, t]$ is CR. For this we will model E as ideal.

Lemma 4.2 *Let $\mathsf{E} : \{0,1\}^\ell \times \{0,1\}^n \to \{0,1\}^n$ be a blockcipher which we model as ideal, and let $1 \le t < n$. Let $\mathsf{G} = \mathbf{ER}[\mathsf{E}, t]$. For any adversary $\mathcal{A}_\mathsf{G}$ making at most $q \le 2^{n-t}$ ideal-cipher queries*

Game G

$\mathrm{E}(L, A)$:
1 If $\mathrm{TE}[L, A] \neq \perp$ then return $\mathrm{TE}[L, A]$
2 $B \leftarrow\!\!\$\ \{0,1\}^n \setminus \mathcal{S}(L)$
3 $\mathcal{P}(L) \leftarrow \mathcal{P}(L) \cup \{A\}$; $\mathcal{S}(L) \leftarrow \mathcal{S}(L) \cup \{B\}$
4 $\mathrm{TE}[L, A] \leftarrow B$; $\mathrm{TI}[L, B] \leftarrow A$
5 Return B

$\mathrm{E}^{-1}(L, B)$:
6 If $\mathrm{TI}[L, B] \neq \perp$ then return $\mathrm{TI}[L, B]$
7 $A \leftarrow\!\!\$\ \{0,1\}^n \setminus \mathcal{P}(L)$
8 $\mathcal{P}(L) \leftarrow \mathcal{P}(L) \cup \{A\}$; $\mathcal{S}(L) \leftarrow \mathcal{S}(L) \cup \{B\}$
9 $\mathrm{TE}[L, A] \leftarrow B$; $\mathrm{TI}[L, B] \leftarrow A$
10 Return A

FINALIZE$(L_1 \| X_1, L_2 \| X_2)$:
11 $B_1 \leftarrow \mathrm{E}(L_1, X_1 \| 0^{n-t})$; $B_2 \leftarrow \mathrm{E}(L_2, X_2 \| 0^{n-t})$
12 Return ($B_1 = B_2$ and $(L_1, X_1) \neq (L_2, X_2)$)

Figure 6: First game for proof of Lemma 4.1.

we have

$$\mathbf{Adv}^{\mathrm{cr}}_{\mathsf{G}}(\mathcal{A}_{\mathsf{G}}) \leq \frac{q}{2^{n-t} \cdot (1 - 2^{-t})} + \frac{(q+2)(q+1)}{2^{n+1}} \ . \tag{2}$$

A related construction is considered in [5, Proposition 5.4]. We give a somewhat better bound and a direct, game-playing proof. (The bound in the statement of [5, Proposition 5.4] may at first glance look better but it has a typo: the 4^s term should be 2^s, as shown in their proof in [5, Appendix G].) Khairallah [30, Proposition 1] shows that an ideal PRI is CR if the output length is sufficiently larger than the input length. We can't use that here since G above is not an ideal PRI, but the closeness of the bounds indicates that G has close to ideal behavior.

Proof of Lemma 4.2: Consider the game G of Figure 6. It starts from game $\mathbf{G}^{\mathrm{cr}}_{\mathsf{G}}$ but models E as ideal by providing an ideal cipher oracle E and its inverse E^{-1}. $\mathrm{TE}[L, A]$ is the value of the cipher on key L and input A, while $\mathrm{TI}[L, B]$ is the value of the inverse cipher on key L and output B. The tables TE, TI are assumed initially $\perp$ everywhere, and are populated in the games via lazy sampling. The sets $\mathcal{P}(L), \mathcal{S}(L)$, assumed initially empty, are used to ensure that $\mathrm{TE}[L, \cdot]$ is a permutation with inverse permutation $\mathrm{TI}[L, \cdot]$. In FINALIZE, the ideal cipher is invoked to compute B_1, B_2, and then the game returns true iff a collision has occurred. We have

$$\mathbf{Adv}^{\mathrm{cr}}_{G}(\mathcal{A}) = \Pr[\mathrm{G}(\mathcal{A})] \ .$$

Next, consider the games in Fig. 7, where G_1 includes the boxed code and G_0 does not. These games define the set $\mathcal{Z}$ at line 1 and then add conditions to set a win flag at lines 5, 12. They also add conditions to set a bad flag at line 10. We first claim that

$$\Pr[\mathrm{G}(\mathcal{A})] \leq \Pr[\mathrm{G}_0(\mathcal{A})] \ .$$

To justify the above claim we need to show that whenever adversary $\mathcal{A}$ wins in G then it also wins in G_0, that is, the win flag is set to true. So suppose $\mathcal{A}$ wins in G, meaning it outputs distinct $L_1 \| X_1, L_2 \| X_2$ such that $\mathrm{E}(L_1, X_1 \| 0^{n-t}) = \mathrm{E}(L_2, X_2 \| 0^{n-t})$. But lines 5,12 set win if any query of $\mathcal{A}$ results in a collision, and line 15 ensures that queries $\mathrm{E}(L_1, X_1 \| 0^{n-t}), \mathrm{E}(L_2, X_2 \| 0^{n-t})$ are made

Game G_0, $\boxed{\mathrm{G}_1}$

INITIALIZE:

1 $\mathcal{Z} \leftarrow \{\, X \| 0^{n-t} \,:\, X \in \{0,1\}^t \,\}$ // $\mathcal{Z}$ is a set of size 2^t

$\mathrm{E}(L, A)$:

2 If $\mathrm{TE}[L, A] \neq \perp$ then return $\mathrm{TE}[L, A]$

3 $B \xleftarrow{\$} \{0,1\}^n \setminus \mathcal{S}(L)$

4 $\mathcal{P}(L) \leftarrow \mathcal{P}(L) \cup \{A\}$; $\mathcal{S}(L) \leftarrow \mathcal{S}(L) \cup \{B\}$

5 If ($(A \in \mathcal{Z})$ and $(\exists (L', A') \in \{0,1\}^\ell \times \mathcal{Z} : \mathrm{TE}[L', A'] = B)$) then $\mathsf{win} \leftarrow \mathsf{true}$

6 $\mathrm{TE}[L, A] \leftarrow B$; $\mathrm{TI}[L, B] \leftarrow A$

7 Return B

$\mathrm{E}^{-1}(L, B)$:

8 If $\mathrm{TI}[L, B] \neq \perp$ then return $\mathrm{TI}[L, B]$

9 $A \xleftarrow{\$} \{0,1\}^n \setminus \mathcal{P}(L)$

10 If $A \in \mathcal{Z}$ then $\mathsf{bad} \leftarrow \mathsf{true}$; $\boxed{A \xleftarrow{\$} \{0,1\}^n \setminus (\mathcal{P}(L) \cup \mathcal{Z})}$

11 $\mathcal{P}(L) \leftarrow \mathcal{P}(L) \cup \{A\}$; $\mathcal{S}(L) \leftarrow \mathcal{S}(L) \cup \{B\}$

12 If ($(A \in \mathcal{Z})$ and $(\exists (L', A') \in \{0,1\}^\ell \times \mathcal{Z} : \mathrm{TE}[L', A'] = B)$) then $\mathsf{win} \leftarrow \mathsf{true}$

13 $\mathrm{TE}[L, A] \leftarrow B$; $\mathrm{TI}[L, B] \leftarrow A$

14 Return A

FINALIZE$(L_1 \| X_1, L_2 \| X_2)$:

15 $B_1 \leftarrow \mathrm{E}(L_1, X_1 \| 0^{n-t})$; $B_2 \leftarrow \mathrm{E}(L_2, X_2 \| 0^{n-t})$

16 Return win

Figure 7: Subsequent games for proof of Lemma 4.1.

---

in the game, so win is set in particular if $L_1 \| X_1, L_2 \| X_2$ form a collision. (But it could be set more often.) Note that lines 5,12 capture non-trivial collisions, as desired, by delaying the settings of TE, TI to after win is set.

Now, games $\mathrm{G}_0, \mathrm{G}_1$ are identical-until-bad [10] so using Lemma 2.1 we have

$$\begin{aligned}\Pr[\mathrm{G}_0(\mathcal{A})] &= \Pr[\mathrm{G}_1(\mathcal{A})] + \Pr[\mathrm{G}_0(\mathcal{A})] - \Pr[\mathrm{G}_1(\mathcal{A})] \\ &\leq \Pr[\mathrm{G}_1(\mathcal{A})] + \Pr[\mathrm{G}_0(\mathcal{A}) \text{ sets } \mathsf{bad}\,] \,.\end{aligned}$$

In the execution of G_0 with $\mathcal{A}$, there are at most $q+2$ oracle queries, the extra 2 due to line 15. We are interested in how often bad is set at line 10. This can only happen with queries to E^{-1}, so there are at most q candidate queries. In the i-th query, A at line 9 is chosen from a set of size at least $2^n - (i-1)$ and so bad is set with probability at most $2^t/(2^n - (i-1))$. Thus by the union bound we have

$$\Pr[\mathrm{G}_0(\mathcal{A}) \text{ sets } \mathsf{bad}\,] \leq \sum_{i=1}^{q} \frac{2^t}{2^n - (i-1)} \leq \frac{q \cdot 2^t}{2^n - (q-1)} \,.$$

The Lemma assumes $q \leq 2^{n-t}$ so $2^n - (q-1) > 2^n - 2^{n-t} = 2^n \cdot (1 - 2^{-t})$, so

$$\frac{q \cdot 2^t}{2^n - (q-1)} \leq \frac{q \cdot 2^t}{2^n \cdot (1 - 2^{-t})} = \frac{q}{2^{n-t} \cdot (1 - 2^{-t})} \,.$$

Next we claim that

$$\Pr[\mathrm{G}_1(\mathcal{A})] \leq \binom{q+2}{2} \cdot \frac{1}{2^n} = \frac{(q+2)(q+1)}{2^{n+1}} \,. \tag{3}$$

Due to the boxed code at line 10, which is included in G_1, we know that the condition on line 12 will never be satisfied. Thus, $\Pr[\mathrm{G}_1(\mathcal{A})]$ is bounded by the probability of setting win at line 5. This can only happen if $\mathrm{TE}(L, A) = \mathrm{TE}[L', A']$ for some, distinct pairs $(L, A), (L', A') \in \{0,1\}^\ell \times \mathcal{Z}$. But then, since E is a block cipher, the collision can only happen if $L \neq L'$. Thus the chance of a collision is 2^{-n}. There are a total of at most $q + 2$ queries (the extra 2 due to line 15) so, by the union bound, we get Eq. (3). Putting everything above together concludes the proof. ▌ Using

Lemmas 4.1 and 4.2 we obtain the following, which establishes the collision-resistance security of the $\mathbf{HER}[\mathsf{H}, \mathsf{E}, t]$ TCC under the assumption that H is collision resistant and E is ideal.

Theorem 4.3 (HER is Collision Resistant) *Let $\mathsf{H} : \{0,1\}^k \times \{0,1\}^* \to \{0,1\}^\ell$ be a function, let $\mathsf{E} : \{0,1\}^\ell \times \{0,1\}^n \to \{0,1\}^n$ be a blockcipher that we model as ideal, and let $1 \leq t < n$. Let $\mathsf{F} = \mathbf{HER}[\mathsf{H}, \mathsf{E}, t]$. For any adversary $\mathcal{A}_\mathsf{F}$ making at most $q \leq 2^{n-t}$ ideal cipher queries, we can construct an adversary $\mathcal{A}_\mathsf{H}$ such that*

$$\mathbf{Adv}^{\mathrm{cr}}_{\mathsf{F}}(\mathcal{A}_\mathsf{F}) \leq \mathbf{Adv}^{\mathrm{cr}}_{\mathsf{H}}(\mathcal{A}_\mathsf{H}) + \frac{q}{2^{n-t} \cdot (1 - 2^{-t})} + \frac{(q+1)(q+2)}{2^{n+1}} \ . \tag{4}$$

The running time of $\mathcal{A}_\mathsf{H}$ is similar to that of $\mathcal{A}_\mathsf{F}$.

The assumption $q \leq 2^{n-t}$ in the Theorem statement may look like a restriction, and a reviewer of a prior version asked if it could be relaxed to $q \leq 2^{n-1}$. Yes, but a term like the second one, meaning roughly of the form $q/2^{n-t}$, would still remain in the bound, and this term is almost one already at $q = 2^{n-t}$, so there is no particular gain to allowing q larger than this.

PRI Security. The following theorem shows that the TCC $\mathsf{F} = \mathbf{HER}[\mathsf{H}, \mathsf{E}, t]$ is PRI secure assuming the underlying function H is a PRF and block cipher E is a strong PRP. Note that this result is in the standard model, and makes no idealized assumptions, meaning we assume neither that H is a random oracle nor that E is an ideal cipher.

Theorem 4.4 (HER is PRI-secure) *Let $\mathsf{H} : \{0,1\}^k \times \{0,1\}^* \to \{0,1\}^\ell$ be a function, let $\mathsf{E} : \{0,1\}^\ell \times \{0,1\}^n \to \{0,1\}^n$ be a block cipher and let $1 \leq t < n$. Let $\mathsf{F} = \mathbf{HER}[\mathsf{H}, \mathsf{E}, t]$. Let $\mathcal{A}_\mathsf{F}$ be an adversary making q_e Enc queries and q_i Inv queries, with at most Q tweaks used across all queries. Then we can construct adversaries $\mathcal{A}_\mathsf{H}$ and $\mathcal{A}_\mathsf{E}$ such that*

$$\mathbf{Adv}^{\mathrm{pri}}_{\mathsf{F}}(\mathcal{A}_\mathsf{F}) \leq \mathbf{Adv}^{\mathrm{prf}}_{\mathsf{H}}(\mathcal{A}_\mathsf{H}) + \mathbf{Adv}^{\pm\mathrm{prp}}_{\mathsf{E}}(\mathcal{A}_\mathsf{E}) \ . \tag{5}$$

Adversary $\mathcal{A}_\mathsf{F}$ makes one New query and Q Fn queries. Adversary $\mathcal{A}_\mathsf{E}$ makes Q New queries, q_e Enc queries and q_i Dec queries. The running times of $\mathcal{A}_\mathsf{H}$ and $\mathcal{A}_\mathsf{E}$ are similar to that of $\mathcal{A}_\mathsf{F}$.

We note that Eq. (5) has no added terms with denominators 2^t or 2^{n-t}. Indeed, the PRI security of $\mathbf{HER}[\mathsf{H}, \mathsf{E}, t]$ does not depend on t. (Its CR security however does.) This is a benefit of our having defined PRI security via pseudorandom injections.

Proof of Theorem 4.4: Consider the games of Figure 8. The $\textsc{Finalize}(b')$ procedure for all these games returns $[[b' = 1]]$ and is omitted for succinctness. For brevity let $\epsilon_i = \Pr[\mathrm{G}_i(\mathcal{A}_\mathsf{F})]$ for $i = 0, \ldots, 3$. First consider games G_0 and G_3. Game G_0 is the real ($b = 1$) case of game $\mathbf{G}^{\mathrm{pri}}_{\mathsf{F}}$ and G_3 is the ideal ($b = 0$) case of game $\mathbf{G}^{\mathrm{pri}}_{\mathsf{F}}$. Hence we have

$$\mathbf{Adv}^{\mathrm{pri}}_{\mathsf{F}}(\mathcal{A}_\mathsf{F}) = \epsilon_0 - \epsilon_3 = (\epsilon_0 - \epsilon_1) + (\epsilon_1 - \epsilon_2) + (\epsilon_2 - \epsilon_3) \ .$$

Game G_0	Game G_1
INITIALIZE: 1 $CK \leftarrow\!\!\$\ \{0,1\}^k$	INITIALIZE: 1 $f \leftarrow\!\!\$\ \mathrm{FUNC}(\{0,1\}^*, \{0,1\}^\ell)$
FN(T, D): 2 $L \leftarrow \mathsf{H}(CK, T)$ 3 $Y \leftarrow \mathsf{E}(L, D\|0^{n-t})$ 4 Return Y	FN(T, D): 2 $L \leftarrow f(T)$ 3 $Y \leftarrow \mathsf{E}(L, D\|0^{n-t})$ 4 Return Y
INV(T, Y): 5 $L \leftarrow \mathsf{H}(CK, T)$ 6 $C \leftarrow \mathsf{E}^{-1}(L, Y)$ 7 $D\|P \leftarrow C$ // $\|P\| = n - t$ 8 If $(P = 0^{n-t})$ then return D 9 Return $\perp$	INV(T, Y): 5 $L \leftarrow f(T)$ 6 $C \leftarrow \mathsf{E}^{-1}(L, Y)$ 7 $D\|P \leftarrow C$ // $\|P\| = n - t$ 8 If $(P = 0^{n-t})$ then return D 9 Return $\perp$

Game G_2	Games G_3
INITIALIZE: 1 For $T \in \{0,1\}^*$ do 2 $\quad \pi_T \leftarrow\!\!\$\ \mathrm{PERM}(n)$	INITIALIZE: 1 For $T \in \{0,1\}^*$ do 2 $\quad I_T \leftarrow\!\!\$\ \mathrm{INJ}(t, n)$
FN(T, D): 3 $Y \leftarrow \pi_T(D\|0^{n-t})$ 4 Return Y	FN(T, D): 3 $Y \leftarrow I_T(D)$ 4 Return Y
INV(T, Y): 5 $C \leftarrow \pi_T^{-1}(Y)$ 6 $D\|P \leftarrow C$ // $\|P\| = n - t$ 7 If $(P = 0^{n-t})$ then return D 8 Return $\perp$	INV(T, Y): 5 Return $I_T^{-1}(Y)$

Figure 8: Games for proof of Theorem 4.4.

We now build adversaries $\mathcal{A}_{\mathsf{H}}, \mathcal{A}_{\mathsf{E}}$ such that

$$\epsilon_0 - \epsilon_1 \leq \mathbf{Adv}^{\mathrm{prf}}_{\mathsf{H}}(\mathcal{A}_{\mathsf{H}}) \tag{6}$$

$$\epsilon_1 - \epsilon_2 \leq \mathbf{Adv}^{\pm\mathrm{prp}}_{\mathsf{E}}(\mathcal{A}_{\mathsf{E}}) \tag{7}$$

$$\epsilon_2 - \epsilon_3 = 0\ . \tag{8}$$

Game G_1 replaces game G_0's calls to $\mathsf{H}(CK, \cdot)$ with calls to a random function f. Adversary $\mathcal{A}_{\mathsf{H}}$ makes a single NEW query, to initialize one user, and then runs $\mathcal{A}_{\mathsf{F}}$, answering oracle queries as per game G_1, except that, at lines 2,5, it computes L by calling its own FN oracle on input $1, T$. It outputs whatever $\mathcal{A}_{\mathsf{F}}$ does. This yields Eq. (6).

Game G_2 replaces game G_1's calls to $\mathsf{E}(f(T), \cdot)$ or $\mathsf{E}^{-1}(f(T), \cdot)$ with calls to π_T and π_T^{-1}, respectively, where π_T is a random permutation, chosen independently for each T at lines 1,2 of G_2. Adversary $\mathcal{A}_{\mathsf{E}}$ runs $\mathcal{A}_{\mathsf{F}}$, answering oracle queries as per game G_2, with the following changes. It identifies each different tweak T with a different user $u(T) \in \mathbb{N}$ in its own game. Then at line 3 of G_2, it answers via its own ENC oracle with inputs $u(T), D\|0^{n-t}$, and likewise at line 5 it answers via its own DEC oracle with inputs $u(T), Y$. The $u(T)$ values are set via a counter c that $\mathcal{A}_{\mathsf{E}}$ maintains, incrementing it, setting $u(T) \leftarrow c$, and calling its NEW oracle, each time it sees a new T. It returns

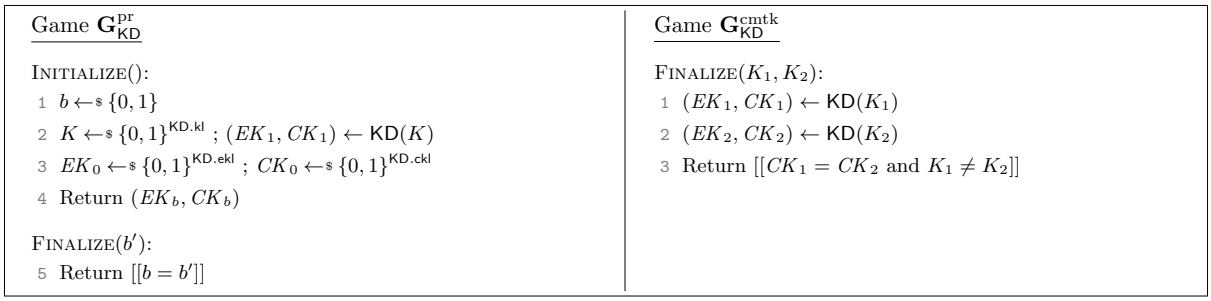


Figure 9: **Left:** Game defining PR security of KDF KD. **Right:** Game defining CMTK security of KDF KD.

---

whatever $\mathcal{A}_{\mathsf{F}}$ returns. This yields Eq. (7).

At line 2, game G_3 picks, for each tweak, a random injection I_T, rather than the random permutation π_T of line 2 of G_3. Line 3 of G_3 uses I_T to obtain Y. Likewise, line 5 uses I_T^{-1}. Recall that as per our definition of the inverse of an injection, I_T^{-1} will return $\perp$ on inputs Y not in the image of I_T. Taking this into account, we can see that the distributions of answers to Fn, Inv queries in G_3 are actually identical to those in G_2. This yields Eq. (8) and completes the proof. ▮

5 The UCX universal committing transform

Here we specify, and prove correct, a committing transform we call **UCX**. It takes an authenticated encryption scheme SE and, with the aid of auxiliary primitives (namely, a TCC and a key-derivation function), turns it into a committing authenticated encryption scheme $\overline{\mathsf{SE}}$. It is universal in the sense that (1) SE can be any scheme in the AE5 syntax, so in particular both classical AE1 (AEAD) and AE2 (nonce hiding AE) are covered, (2) It does not assume SE is tag based, and (3) It preserves both unique-nonce and misuse-resistant security. Additionally, the transform is efficient and has small ciphertext expansion.

Key Derivation Functions. The first step of UCX will be to derive, from the main key K, two subkeys, an encryption subkey EK and a commitment subkey CK. This will be done via a key-derivation function (KDF) $\mathsf{KD} : \{0,1\}^{\mathsf{KD.kl}} \to \{0,1\}^{\mathsf{KD.ekl}} \times \{0,1\}^{\mathsf{KD.ckl}}$ with some special properties that we now define. Efficient KDFs meeting our definitions are easily constructed.

In Figure 9, we show games for our two desired security metrics for KDFs. The first, on the left, is a pseudorandomness game which formalizes the goal that output subkeys from a KDF should be indistinguishable from random bit strings. For any adversary $\mathcal{A}$ we let $\mathbf{Adv}^{\mathrm{pr}}_{\mathsf{KD}}(\mathcal{A}) = 2 \cdot \Pr[\mathbf{G}^{\mathrm{pr}}_{\mathsf{KD}}(\mathcal{A})] - 1$. On the right in Figure 9, we give the CMTK game for KDF KD. This asks that the key K commits to the committing subkey CK. An adversary $\mathcal{A}$ playing the game $\mathbf{G}^{\mathrm{cmtk}}_{\mathsf{KD}}$ wins if it can find two unequal input keys K_1, K_2 such that if $(EK_1, CK_1) \leftarrow \mathsf{KD}(K_1)$ and $(EK_2, CK_2) \leftarrow \mathsf{KD}(K_2)$ then $CK_1 = CK_2$. Note that we do not require that the encryption key be committed to. We let $\mathbf{Adv}^{\mathrm{cmtk}}_{\mathsf{KD}}(\mathcal{A}) = \Pr[\mathbf{G}^{\mathrm{cmtk}}_{\mathsf{KD}}(\mathcal{A})]$. KD is easily instantiated, for example via HMAC-SHA512.

UCX. Let SE be an AE5 scheme. Let F be a tweakable committing concealer with $\mathsf{F.Ev} : \{0,1\}^{\mathsf{F.kl}} \times \{0,1\}^* \times \{0,1\}^{\mathsf{F.il}} \to \{0,1\}^{\mathsf{F.ol}}$. Let $\mathsf{KD} : \{0,1\}^{\mathsf{KD.kl}} \to \{0,1\}^{\mathsf{SE.kl}} \times \{0,1\}^{\mathsf{F.kl}}$ be a KDF. (Here $\mathsf{KD.ekl} = \mathsf{SE.kl}$, meaning the encryption subkey will be a key for SE, and $\mathsf{KD.ckl} = \mathsf{F.kl}$, meaning a commitment subkey will be a key for F.) Let $t \leq \mathsf{SE.cl}(0)$ be an integer less than the length of any SE ciphertext. The **UCX** transform builds from KD, F and t the AE5 scheme $\overline{\mathsf{SE}} = \mathbf{UCX}[\mathsf{SE}, \mathsf{F}, \mathsf{KD}, t]$

$\overline{\mathsf{SE}}.\mathsf{Enc}(K, NP, NS, A, M)$:

1 $(EK, CK) \leftarrow \mathsf{KD}(K)$
2 $C \leftarrow \mathsf{SE}.\mathsf{Enc}(EK, NP, NS, A, M)$
3 $C' \| D \leftarrow C \;//\; |D| = t$
4 $R \leftarrow \mathsf{F}.\mathsf{Ev}(CK, NP \| A, D)$
5 $\overline{C} \leftarrow C' \| R$
6 Return $\overline{C}$

$\overline{\mathsf{SE}}.\mathsf{Dec}(K, NP, A, \overline{C})$:

7 $(EK, CK) \leftarrow \mathsf{KD}(K)$
8 $C' \| R \leftarrow \overline{C} \;//\; |R| = \mathsf{F}.\mathsf{ol}$
9 $D \leftarrow \mathsf{F}.\mathsf{Inv}(CK, NP \| A, R)$
10 If $D = \perp$ then return $\perp$
11 $C \leftarrow C' \| D$
12 $(M, NS) \leftarrow \mathsf{SE}.\mathsf{Dec}(EK, NP, A, C)$
13 Return (M, NS)

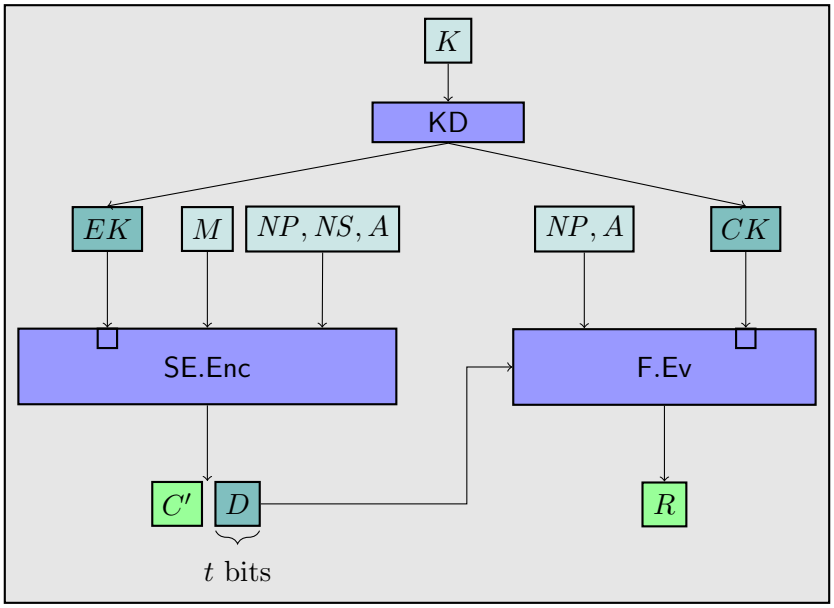


Figure 10: **Top:** Encryption and decryption algorithms of the AE scheme $\overline{\mathsf{SE}} = \mathbf{UCX}[\mathsf{SE}, \mathsf{F}, \mathsf{KD}, t]$ built from our **UCX** transform. **Bottom:** Encryption, pictorially.

---

whose encryption and decryption algorithms are shown in Figure 10, with a picture for the former below. The key-length of $\overline{\mathsf{SE}}$ is $\mathsf{KD}.\mathsf{kl}$. The lengths of public and secret nonces, as well as the AD space and message space, are the same as for SE. The ciphertext length is $\overline{\mathsf{SE}}.\mathsf{cl}(m) = \mathsf{SE}.\mathsf{cl}(m) - t + \mathsf{F}.\mathsf{ol}$, so that the ciphertext expansion is similarly $\overline{\mathsf{SE}}.\mathsf{ce}(m) = \mathsf{SE}.\mathsf{ce}(m) - t + \mathsf{F}.\mathsf{ol}$. The commitment ciphertext overhead is $\mathsf{F}.\mathsf{ol} - t$. A typical instantiation will have $t = 128$ and $\mathsf{F}.\mathsf{ol} = 256$ for 128-bit committing security, in which case the increment in ciphertext size relative to SE (i.e. committing ciphertext expansion) is 128 bits. This is the same as for CTX [14] and CTY [5], even though those only work if SE is tag based.

UCY: An optimization for UNAE. We remark that if the goal is only to preserve UNAE (rather than both UNAE and MRAE) then an optimization is possible. Namely, just as CTY [5] optimized CTX [14] by replacing A by ε in encryption with the base scheme, **UCX** can also replace A by ε at lines 2,12 of Figure 10, which increases efficiency for the UNAE case. We call this transform **UCY**. We warn however that a simple attack shows that **UCY** does *not* preserve MRAE; for that, we need the full **UCX** of Figure 10.

UCX preserves tag-basedness. At line 3 of Fig. 10, the SE-ciphertext C is split to let D be the last t bits. **UCX** does not require that the starting scheme SE is tag based, so in general D is simply the ciphertext suffix, but if SE happens to be tag based, it may be convenient to let

$t = \mathsf{SE.tl}$, in which case D is the tag. A nice property of **UCX**, given in the following Proposition, is that, in this case, $\overline{\mathsf{SE}}$ will also be tag based. So one can for example apply the SC transform of [5] to shorten the $\overline{\mathsf{SE}}$ ciphertext.

Proposition 5.1 *Let* SE *be a tag-based symmetric encryption scheme. Let* $t = \mathsf{SE.tl}$ *and let* $\overline{\mathsf{SE}} = \mathbf{UCX}[\mathsf{SE}, \mathsf{F}, \mathsf{KD}, t]$. *Then* $\overline{\mathsf{SE}}$ *is also tag based, with* $\mathsf{SE.tl} = \mathsf{F.ol}$.

Proof of Proposition 5.1: The tagging algorithm for $\overline{\mathsf{SE}}$ takes K, NP, A, C'. It lets $(EK, CK) \leftarrow \mathsf{KD}(K)$ and runs the tagging algorithm of SE on EK, NP, A, C' to get (M, NS, T^*). If this is $\perp$ then it returns $\perp$. Else it lets $R^* \leftarrow \mathsf{F}(CK, NP \| A, T^*)$ and returns (M, NS, R^*). ▮

CMT SECURITY. The following theorem establishes CMT security of $\overline{\mathsf{SE}} = \mathbf{UCX}[\mathsf{SE}, \mathsf{F}, \mathsf{KD}, t]$ assuming CR security of F and CMTK security of KD. We note that there is a subtle element here namely that, despite not being an input to $\mathsf{F.Ev}$, the secret nonce is committed to by the ciphertext. Intuitively this is true because decryption returns the secret nonce.

Theorem 5.2 (UCX provides committing security) *Let* $\overline{\mathsf{SE}} = \mathbf{UCX}[\mathsf{SE}, \mathsf{F}, \mathsf{KD}, t]$ *be the symmetric encryption scheme built from our* **UCX** *transform from symmetric encryption scheme* SE, *tweakable committing concealer* F, *key-derivation function* KD *and integer* t. *Given adversary* $\mathcal{A}$ *we can construct adversaries* $\mathcal{B}$ *and* $\mathcal{D}$ *such that*

$$\mathbf{Adv}^{\mathrm{cmt}}_{\overline{\mathsf{SE}}}(\mathcal{A}) \leq \mathbf{Adv}^{\mathrm{cr}}_{\mathsf{F}}(\mathcal{B}) + \mathbf{Adv}^{\mathrm{cmtk}}_{\mathsf{KD}}(\mathcal{D}) \ . \tag{9}$$

The running times of $\mathcal{B}, \mathcal{D}$ *are about that of* $\mathcal{A}$.

Proof of Theorem 5.2: Let (T_1, T_2) be the output of $\mathcal{A}$, where, $T_i = (K_i, NP_i, NS_i, A_i, M_i)$ for $i \in \{1, 2\}$. Now for $i \in \{1, 2\}$, let

$(EK_i, CK_i) \leftarrow \mathsf{KD}(K_i)$; $C'_i \| D_i \leftarrow \mathsf{SE.Enc}(K_i, NP_i, NS_i, A_i, M_i)$
$R_i \leftarrow \mathsf{F.Ev}(CK_i, NP_i \| A_i, D_i)$

Let EV be the event that $C'_1 \| R_1 = C'_2 \| R_2$ and $T_1 \neq T_2$. Let EV_1 be the event that $(CK_1, NP_1, A_1, D_1) \neq (CK_2, NP_2, A_2, D_2)$. Then

$$\mathbf{Adv}^{\mathrm{cmt}}_{\overline{\mathsf{SE}}}(\mathcal{A}) = \Pr[\mathrm{EV}] = \Pr[\mathrm{EV} \cap \mathrm{EV}_1] + \Pr[\mathrm{EV} \cap \overline{\mathrm{EV}_1}] \ .$$

We construct adversary $\mathcal{B}$ so that $\Pr[\mathrm{EV} \cap \mathrm{EV}_1] \leq \mathbf{Adv}^{\mathrm{cr}}_{\mathsf{F}}(\mathcal{B})$. Adversary $\mathcal{B}$ simply runs $\mathcal{A}$ to get $((K_1, NP_1, NS_1, A_1, M_1), (K_2, NP_2, NS_2, A_2, M_2))$ and outputs $((CK_1, NP_1 \| A_1, D_1), (CK_2, NP_2 \| A_2, D_2))$, which are computed as above. If $\mathrm{EV} \cap \mathrm{EV}_1$ holds then $\mathcal{B}$ clearly outputs a non-trivial collision for $\mathsf{F.Ev}$ and wins its game.

We construct adversary $\mathcal{D}$ so that $\Pr[\mathrm{EV} \cap \overline{\mathrm{EV}_1}] \leq \mathbf{Adv}^{\mathrm{cmtk}}_{\mathsf{KD}}(\mathcal{D})$. It runs $\mathcal{A}$ to get $((K_1, NP_1, NS_1, A_1, M_1), (K_2, NP_2, NS_2, A_2, M_2))$ and outputs (K_1, K_2). To see that this works, assume $\mathrm{EV} \cap \overline{\mathrm{EV}_1}$ holds. This means $(CK_1, NP_1, A_1, D_1) = (CK_2, NP_2, A_2, D_2)$. In particular $CK_1 = CK_2$, so $\mathcal{D}$ wins as long as $K_1 \neq K_2$. We now check the latter. Since EV holds, $C'_1 \| R_1 = C'_2 \| R_2$. Denote this common value by $\overline{C}$. Assume towards a contradiction that $K_1 = K_2$ and let $(M'_i, NS'_i) \leftarrow \overline{\mathsf{SE}}.\mathsf{Dec}(K_i, NP_i, A_i, \overline{C})$ for $i \in \{1, 2\}$. Since $(K_1, NP_1, A_1) = (K_2, NP_2, A_2)$ it must be that $(M'_1, NS'_1) = (M'_2, NS'_2)$. But by correctness of $\overline{\mathsf{SE}}$ it must be that $(M'_i, NS'_i) = (M_i, NS_i)$ for $i \in \{1, 2\}$. Thus we have $(K_1, NP_1, NS_1, A_1, M_1) = (K_2, NP_2, NS_2, A_2, M_2)$. But this contradicts that EV holds, for the latter excluded this. ▮

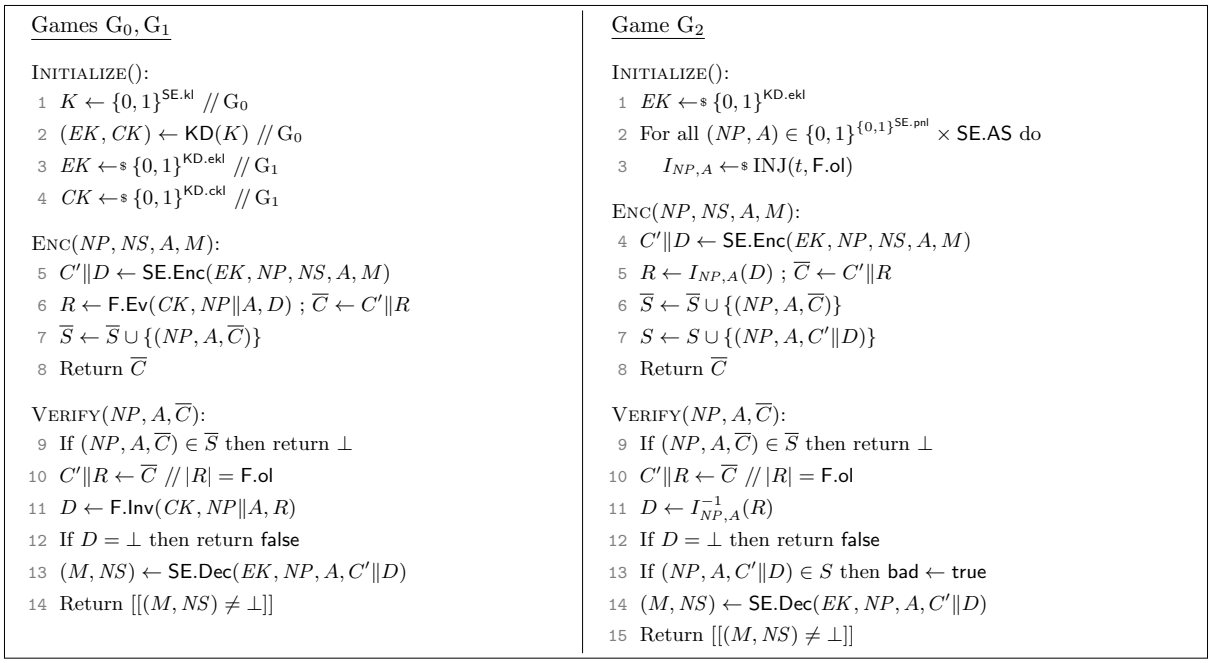


Figure 11: Games for proof of Theorem 5.3.

AE SECURITY. We now show that the **UCX** transform preserves both UNAE and MRAE security. That is, if SE is UNAE secure then so is $\overline{\mathsf{SE}} = \mathbf{UCX}[\mathsf{SE}, \mathsf{KD}, \mathsf{F}, t]$, and likewise if SE is MRAE secure then so is $\overline{\mathsf{SE}} = \mathbf{UCX}[\mathsf{SE}, \mathsf{KD}, \mathsf{F}, t]$. The following Theorem establishes both claims simultaneously assuming pseudorandomness of KD and T-PRI security of the tweakable committing concealer F.

Theorem 5.3 (UCX preserves UNAE and MRAE security) *Let* $\overline{\mathsf{SE}} = \mathbf{UCX}[\mathsf{SE}, \mathsf{F}, \mathsf{KD}, t]$ *be the symmetric encryption scheme built from our* **UCX** *transform from symmetric encryption scheme* SE*, tweakable committing concealer* F*, key-derivation function* KD *and integer* t*. Let* $\mathrm{xx} \in \{\mathrm{un}, \mathrm{mr}\}$*. Let* $\mathcal{A}$ *be an adversary making* q_{e} ENC *queries and* q_{v} VERIFY *queries. Let* $Q_{NP,A}$ *be*

number of $\text{ENC}(NP, \cdot, A, \cdot)$ *queries of* $\mathcal{A}$ *and let* $B = \max_{NP,A} Q_{NP,A}$. *Then we can construct adversaries* $\mathcal{A}_{\mathsf{SE}}$, $\mathcal{A}_{\mathsf{F}}$ *and* $\mathcal{A}_{\mathsf{KD}}$ *such that*

$$\begin{aligned}\mathbf{Adv}^{\text{xx-ae}}_{\overline{\mathsf{SE}}}(\mathcal{A}) \leq & \mathbf{Adv}^{\text{pr}}_{\mathsf{KD}}(\mathcal{A}_{\mathsf{KD}}) + \mathbf{Adv}^{\text{pri}}_{\mathsf{F}}(\mathcal{A}_{\mathsf{F}}) + \mathbf{Adv}^{\text{xx-ae}}_{\mathsf{SE}}(\mathcal{A}_{\mathsf{SE}}) \\ & + \frac{q_{\text{e}} \cdot B}{2^{\mathsf{F.ol}}} + \sum_{NP,A} \frac{Q_{NP,A} \cdot (Q_{NP,A} - 1)}{2^{t+1}} \,. \end{aligned} \tag{10}$$

Adversary $\mathcal{A}_{\mathsf{F}}$ *makes* q_{e} FN *queries and* q_{d} INV *queries. The query profile of* $\mathcal{A}_{\mathsf{SE}}$ *is the same as that of* $\mathcal{A}$. *The constructed adversaries have about the same running time as* $\mathcal{A}$.

In our instantiation $\mathsf{F.ol} = 256$ so the $q_{\text{e}} B \cdot 2^{-\mathsf{F.ol}}$ term in the bound is negligible. With regard to the $\sum_{NP,A} Q_{NP,A} \cdot (Q_{NP,A} - 1) \cdot 2^{-(t+1)}$ term, we note that it is actually 0 in the most popular case in practice, namely that SE is an AE1 (that is, AEAD [37]) scheme and we target UNAE security (meaning xx = un). Indeed, in this case, $\mathsf{SE.snl} = 0$ and $\mathcal{A}$ does not repeat a nonce, so NP values are not repeated across ENC queries, so $Q_{NP,A} \leq 1$ for all (NP, A), and thus $\sum_{NP,A} Q_{NP,A} \cdot (Q_{NP,A} - 1) = 0$. In general, note that $\sum_{NP,A} Q_{NP,A} = q_{\text{e}}$ and hence

$$\sum_{NP,A} \frac{Q_{NP,A} \cdot (Q_{NP,A} - 1)}{2^{t+1}} = \sum_{NP,A} \frac{Q^2_{NP,A}}{2^{t+1}} - \frac{q_{\text{e}}}{2^{t+1}} \leq \frac{1}{2^{t+1}} \cdot \left(\sum_{NP,A} Q_{NP,A} \right)^2 - \frac{q_{\text{e}}}{2^{t+1}} = \frac{q_{\text{e}}(q_{\text{e}} - 1)}{2^{t+1}} \,.$$

Proof of Theorem 5.3: Consider the games of Figures 11 and 12. The $\text{FINALIZE}(b')$ procedure for all these games returns $[[b' = 1]]$ and is omitted for succinctness. Additionally, the ENC procedures, in all games, should begin with the following lines, which have also been dropped for succinctness:

Require: $(NP, NS) \notin U$ // xx = un
$U \leftarrow U \cup \{(NP, NS)\}$ // xx = un
Require: $(NP, NS, A, M) \notin U$ // xx = mr
$U \leftarrow U \cup \{(NP, NS, A, M)\}$ // xx = mr

For brevity let $\epsilon_i = \Pr[\text{G}_i(\mathcal{A})]$ for $i = 0, \ldots, 8$. Game G_0 is the real ($b = 1$) case of game $\mathbf{G}^{\text{xx-ae}}_{\overline{\mathsf{SE}}}$ and game G_8 is the ideal ($b = 0$) case of game $\mathbf{G}^{\text{xx-ae}}_{\overline{\mathsf{SE}}}$. Hence we have

$$\begin{aligned}&\mathbf{Adv}^{\text{xx-ae}}_{\overline{\mathsf{SE}}}(\mathcal{A}) = \epsilon_0 - \epsilon_8 \\ &= (\epsilon_0 - \epsilon_1) + (\epsilon_1 - \epsilon_2) + (\epsilon_2 - \epsilon_3) + (\epsilon_3 - \epsilon_4) + (\epsilon_4 - \epsilon_5) + (\epsilon_5 - \epsilon_8) \,.\end{aligned}$$

We now bound each of the above differences in turn, building adversaries $\mathcal{A}_{\mathsf{SE}}, \mathcal{A}_{\mathsf{F}}, \mathcal{A}_{\mathsf{KD}}$ such that

$$\epsilon_0 - \epsilon_1 \leq \mathbf{Adv}^{\text{pr}}_{\mathsf{KD}}(\mathcal{A}_{\mathsf{KD}}) \tag{11}$$

$$\epsilon_1 - \epsilon_2 \leq \mathbf{Adv}^{\text{pri}}_{\mathsf{F}}(\mathcal{A}_{\mathsf{F}}) \tag{12}$$

$$\epsilon_2 - \epsilon_3 \leq \mathbf{Adv}^{\text{xx-ae}}_{\mathsf{SE}}(\mathcal{A}_{\mathsf{SE}}) \tag{13}$$

$$\epsilon_3 - \epsilon_4 = 0 \tag{14}$$

$$\epsilon_4 - \epsilon_5 \leq \frac{q_{\text{e}} \cdot B}{2^{\mathsf{F.ol}}} \tag{15}$$

$$\epsilon_5 - \epsilon_8 \leq \textstyle\sum_{NP,A} \dfrac{Q_{NP,A} \cdot (Q_{NP,A} - 1)}{2^{t+1}} \,. \tag{16}$$

From G_0 to G_1, the only change is to switch (EK, CK) from being generated via KD (lines 1,2) to being chosen at random (lines 3,4). Adversary $\mathcal{A}_{\mathsf{KD}}$ has input EK, CK from $\mathbf{G}^{\mathrm{pr}}_{\mathsf{KD}}$.Initialize. It runs $\mathcal{A}$, answering queries as in G_1, except that it uses its own input keys at lines 5,6,11,13. It outputs as its decision whatever b' adversary $\mathcal{A}$ outputs. This yields Eq. (11).

Lines 2,3 of game G_2 pick a random injection per (NP, A) pair, viewing this pair as a tweak from the point of view of the TCC F. Game G_1's use of $\mathsf{F.Ev}(CK, NP\|A, \cdot)$ (line 6) is replaced in G_2 by $I_{NP,A}(\cdot)$ (line 5). Likewise, the use of $\mathsf{F.Inv}(CK, NP\|A, \cdot)$ (line 11 in G_1) is replaced by $I^{-1}_{NP,A}(\cdot)$ (line 11 in G_2). Adversary $\mathcal{A}_{\mathsf{F}}$ itself picks $EK \leftarrow\!\!\$\ \{0,1\}^{\mathsf{KD.ekl}}$ and then runs $\mathcal{A}$, answering queries as in G_2, with the following changes. At line 5, it invokes its own oracle to set $R \leftarrow \mathbf{G}^{\mathrm{pri}}_{\mathsf{F}}.\textsc{Fn}(NP\|A, D)$ and at line 11 likewise sets $D \leftarrow \mathbf{G}^{\mathrm{pri}}_{\mathsf{F}}.\textsc{Inv}(NP\|A, R)$. It outputs as its decision whatever b' adversary $\mathcal{A}$ outputs. This yields Eq. (12).

Game G_2 uses $\mathsf{SE.Enc}$ (line 4) to determine $C'\|D$. Game G_3 (line 3) instead picks $C'\|D$ at random. Oracle G_3.Verify skips the checks made in G_2.Verify and in their place just returns false. Conceptually, adversary $\mathcal{A}_{\mathsf{SE}}$ begins by executing lines 2,3 of G_2 to pick the injections $I_{NP,A}$. (In actuality, for efficiency, it will have to sample them lazily.) It then runs $\mathcal{A}$, answering queries as in G_2, with the following changes. At line 4, it invokes its own oracle to set $C'\|D \leftarrow \mathbf{G}^{\text{xx-ae}}_{\mathsf{SE}}.\textsc{Enc}(NP, NS, A, M)$. In place of lines 14, 15, it calls $d \leftarrow \mathbf{G}^{\text{xx-ae}}_{\mathsf{SE}}.\textsc{Verify}(NP, A, C'\|D)$ and returns d. It outputs as its decision whatever b' adversary $\mathcal{A}$ outputs. There are several things to check to verify Eq. (13). Let b denote the challenge bit in game $\mathbf{G}^{\text{xx-ae}}_{\mathsf{SE}}$ and b' the output of $\mathcal{A}_{\mathsf{SE}}$ in this game. Then our first claim is that, when $b = 1$, adversary $\mathcal{A}_{\mathsf{SE}}$ is properly simulating the execution of $\mathcal{A}$ in G_2, meaning

$$\Pr[\, b' = 1 \,|\, b = 1 \,] = \Pr[\mathrm{G}_2(\mathcal{A})] \,. \tag{17}$$

To justify this first notice that the simulation of the encryption oracle is correct in both the xx = un and xx = mr cases because $\mathcal{A}_{\mathsf{SE}}$ is playing game $\mathbf{G}^{\text{xx-ae}}_{\mathsf{SE}}$ so the "Require" conditions are met. Now a subtle point with Verify queries is that for the simulation to be correct we require that bad is not set at line 13, since otherwise $\mathcal{A}_{\mathsf{SE}}$'s call to $\mathbf{G}^{\text{xx-ae}}_{\mathsf{SE}}.\textsc{Verify}$ will return $\perp$. Indeed we claim that the probability of setting bad is 0. For this we make crucial use of the injectivity of $I_{NP,A}$. Consider a $\textsc{Verify}(NP, A, C'\|R)$ query for which $(NP, A, C'\|R) \notin \overline{S}$ at line 9 of G_2, and also $D \neq \perp$ at line 12, so that the test of line 13 is executed. We claim this test will be negative, meaning $(NP, A, C'\|D)$ will not be in S. Suppose towards a contradiction that $(NP, A, C'\|D) \in S$ at line 13. So there was a $\textsc{Enc}(NP, NS, A, M)$ query that put $(NP, A, C'\|D)$ in S at line 7. So $(NP, A, C'\|R^*)$ was put in $\overline{S}$ at line 6 where $R^* = I_{NP,A}(D)$. But $D = I^{-1}_{NP,A}(R)$ (line 11), so the injectivity of $I_{NP,A}$ means that $R = R^*$, and thus $(NP, A, C'\|R) \in \overline{S}$ at line 9, which is the desired contradiction. This verifies Eq. (17). Now to conclude Eq. (13), we also have to show that

$$\Pr[\, b' = 1 \,|\, b = 0 \,] = \Pr[\mathrm{G}_3(\mathcal{A})] \,. \tag{18}$$

To justify this first note that the simulation of the encryption oracle is correct because $\mathbf{G}^{\text{xx-ae}}_{\mathsf{SE}}.\textsc{Enc}$ returns random ciphertexts when $b = 0$. Now, lines 9–13 of G_3 just end up returning false. We want to say this is also what $\mathbf{G}^{\text{xx-ae}}_{\mathsf{SE}}.\textsc{Dec}$ returns when $b = 0$, so the simulation is correct, but there is again a caveat, namely that this is only true if bad is not set at line 12 of G_3, since otherwise $\mathbf{G}^{\text{xx-ae}}_{\mathsf{SE}}.\textsc{Dec}$ will return $\perp$. However, the argument that bad is never set at line 12 is the same as above, so we have Eq. (18) and thus Eq. (13).

Lines 9–13 of G_3 simply return false, so Verify can be simplified to the one of G_4. This justifies Eq. (14).

The change from G_4 to G_5 (Figure 11) is only that line 2 is replaced with line 3 so $I_{NP,A}$ is now

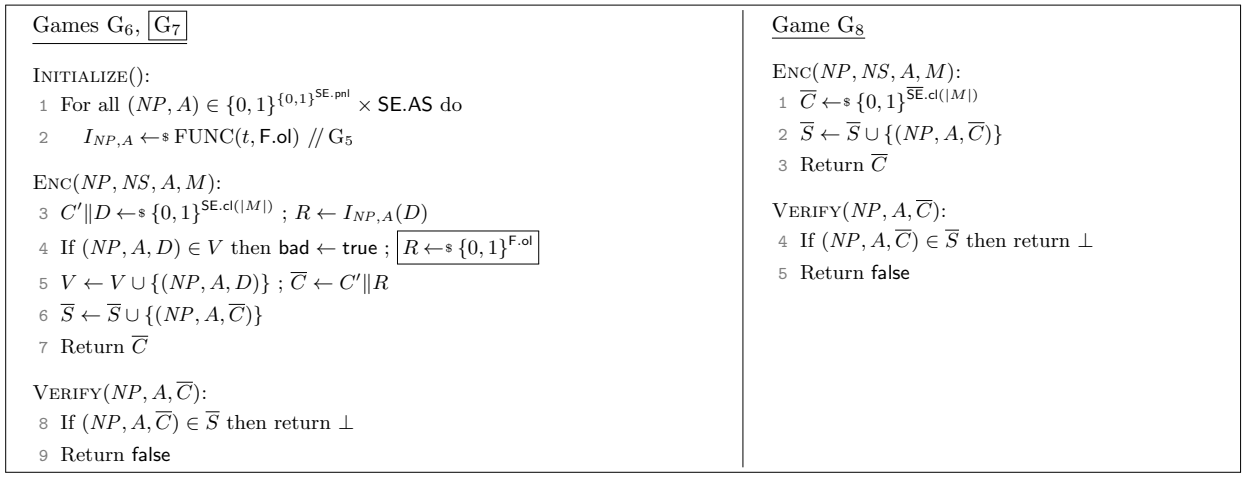


Figure 12: More games for proof of Theorem 5.3.

---

a random function rather than a random injection. Now note that these games do not use $I^{-1}_{NP,A}$. This allows us, referring to the games of Fig. 2, to build an adversary $\mathcal{B}$ such that

$$\epsilon_4 - \epsilon_5 \leq \Pr[\mathbf{G}^{\mathrm{ri}}_{t,n}(\mathcal{B})] - \Pr[\mathbf{G}^{\mathrm{rf}}_{t,n}(\mathcal{B})] \ .$$

The idea is that adversary $\mathcal{B}$ sees the different tweaks $NP\|A$ in games G_4, G_5 as different users. It makes q_{e} FN queries with at most B queries per user, so we can apply our Multi-user RI/RF switching lemma (Lemma 2.2) to get

$$\Pr[\mathbf{G}^{\mathrm{ri}}_{t,n}(\mathcal{B})] - \Pr[\mathbf{G}^{\mathrm{rf}}_{t,n}(\mathcal{B})] \leq \frac{q_{\mathrm{e}} \cdot B}{2^{\mathsf{F.ol}}} \ .$$

This yields Eq. (15).

We want that the distribution of $\overline{C}$ returned by ENC be uniform. This is not true in G_5. To see this consider two different ENC queries that have the same NP and A, but, say, different M. Now, if the D values at line 4 are the same for these queries, then the R values will be the same as well. To transition to random $\overline{C}$, we consider the games of Figure 12. Game G_7 includes the boxed code, and G_6 does not. So G_6 is equivalent to G_5. However, G_6 and G_5 are identical-until-bad, so we can use the Fundamental Lemma of Game Playing [10] to get

$$\begin{aligned} \epsilon_5 - \epsilon_7 = \epsilon_6 - \epsilon_7 &\leq \Pr[\mathrm{G}_6(\mathcal{A}) \text{ sets } \mathsf{bad}\,] \\ &\leq \sum_{NP,A} \sum_{i=1}^{Q_{NP,A}-1} \frac{i}{2^{t+1}} = \sum_{NP,A} \frac{Q_{NP,A} \cdot (Q_{NP,A} - 1)}{2^{t+1}} \ . \end{aligned} \tag{19}$$

Above we used the fact that D has length t and is chosen at random at line 3 of G_6. We also used the definition of $Q_{NP,A}$ from the Theorem Statement.

Game G_7 includes the boxed code, so $\overline{C}$ at line 7 is uniformly distributed, as in G_8, so we have $\epsilon_7 = \epsilon_8$. Combining this with Eq. (19), we have Eq. (16), which concludes the proof. ∎

BANDWITH OVERHEAD AND INSTANTIATION. Let $\overline{\mathsf{SE}} = \mathbf{UCX}[\mathsf{SE}, \mathsf{F}, \mathsf{KD}, t]$ with $\mathsf{F} = \mathbf{HER}[\mathsf{H}, \mathsf{E}, t]$ and let n be the block length of E. Since CMT security is subject to offline attacks, $s = 128$ bits of it are advocated. To achieve this our suggested instantiation is $\mathsf{E} = \mathsf{Rijndael\text{-}256}$ and $t = 128$, so

that $n = 256$ and $n - t = 128$. For H we could use HMAC-SHA512 with output truncated to 256 bits. Let τ be the ciphertext expansion of SE, and assume it is $\tau = 128$ bits. Then the ciphertext expansion of $\overline{\mathsf{SE}}$ is $\tau + (n - t) = 128 + (256 - 128) = 256$ bits.

However, here we used the conventional measure of ciphertext expansion, namely that it is the length of the ciphertext minus the length of the message. (Also minus the length of the secret nonce, but for simplicity let us assume AE1 so this term is zero.) This, however, will, as above depend on the ciphertext expansion τ of the base scheme SE. So a perhaps more "fair" measure of the bandwidth overhead incurred by **UCX** is what we called commitment ciphertext overhead in Section 1. Recall this was defined as the length of the $\overline{\mathsf{SE}}$-ciphertext minus the length of the SE-ciphertext, and is what is shown in Table 1. Then the commitment overhead of **UCX** is always $n - t$. For the above choice $t = 128$, it is thus 128. This explains the entry for **UCX** in Table 1.

Note that by Theorem 4.3, the number of bits of CMT security provided by **UCX** is $\min(n - t, n/2)$, which is $n - t$ if $t \geq n/2$. In particular for $n = 256$ it is $n - t$ as long as $t \geq 128$. So in this case, the commitment overhead is equal to the number s of bits of CMT security provided.

Reviewers of a prior version of this paper asked what would happen if we wanted only $s = 96$ bits of CMT security. Their worry was that with Rijndael-256 we would have to use the same settings as above, namely $t = 128$, and thus get the same 128 bit commitment overhead. However, we can in fact set $t = 160$ so that $n - t = 256 - 160 = 96$ and thus the commitment ciphertext overhead is just $n - t = 96 = s$. Note since $t = 160 \geq 128 = 256/2 = n/2$, by the above, we do indeed get (about) $n - t = 96$ bits of CMT security.

Throughout, above, we have assumed the base scheme SE to which **UCX** is applied is standard model. If it uses an ideal primitive (such as a random oracle or block cipher) one must pick this to be independent of, and different from, the ideal primitives used by **UCX** itself, using appropriate domain separation as necessary [3]. Once this is done, the proofs extend easily, since the ideal primitive(s) of SE can be directly simulated by adversaries.

Acknowledgments

We thank Rohit Pai for extensive and insightful comments and feedback. We thank the reviewers of Eurocrypt 2026 and Crypto 2026 for their reviews and feedback.

References

[1] Abdalla, M., Bellare, M., Neven, G.: Robust encryption. In: Micciancio, D. (ed.) TCC 2010. LNCS, vol. 5978, pp. 480–497. Springer, Berlin, Heidelberg (Feb 2010) 3

[2] Albertini, A., Duong, T., Gueron, S., Kölbl, S., Luykx, A., Schmieg, S.: How to abuse and fix authenticated encryption without key commitment. In: Butler, K.R.B., Thomas, K. (eds.) USENIX Security 2022. pp. 3291–3308. USENIX Association (Aug 2022), https://www.usenix.org/conference/usenixsecurity22/presentation/albertini 3

[3] Bellare, M., Davis, H., Günther, F.: Separate your domains: NIST PQC KEMs, oracle cloning and read-only indifferentiability. In: Canteaut, A., Ishai, Y. (eds.) EUROCRYPT 2020, Part II. LNCS, vol. 12106, pp. 3–32. Springer, Cham (May 2020) 26

[4] Bellare, M., Hoang, V.T.: Efficient schemes for committing authenticated encryption. In: Dunkelman, O., Dziembowski, S. (eds.) EUROCRYPT 2022, Part II. LNCS, vol. 13276, pp. 845–875. Springer, Cham (May / Jun 2022) 3, 5, 6, 9, 12

[5] Bellare, M., Hoang, V.T.: Succinctly-committing authenticated encryption. In: Reyzin, L., Stebila, D. (eds.) CRYPTO 2024, Part IV. LNCS, vol. 14923, pp. 305–339. Springer, Cham (Aug 2024) 3, 4, 5, 6, 10, 11, 12, 13, 15, 20, 21

[6] Bellare, M., Kilian, J., Rogaway, P.: The security of cipher block chaining. In: Desmedt, Y. (ed.) CRYPTO'94. LNCS, vol. 839, pp. 341–358. Springer, Berlin, Heidelberg (Aug 1994) 9

[7] Bellare, M., Namprempre, C.: Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. In: Okamoto, T. (ed.) ASIACRYPT 2000. LNCS, vol. 1976, pp. 531–545. Springer, Berlin, Heidelberg (Dec 2000) 4, 11

[8] Bellare, M., Ng, R., Tackmann, B.: Nonces are noticed: AEAD revisited. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019, Part I. LNCS, vol. 11692, pp. 235–265. Springer, Cham (Aug 2019) 3, 4, 7, 8, 10

[9] Bellare, M., Rogaway, P.: Encode-then-encipher encryption: How to exploit nonces or redundancy in plaintexts for efficient cryptography. In: Okamoto, T. (ed.) ASIACRYPT 2000. LNCS, vol. 1976, pp. 317–330. Springer, Berlin, Heidelberg (Dec 2000) 4, 14

[10] Bellare, M., Rogaway, P.: The security of triple encryption and a framework for code-based game-playing proofs. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 409–426. Springer, Berlin, Heidelberg (May / Jun 2006) 8, 9, 16, 25

[11] Bellare, M., Rogaway, P., Wagner, D.: The EAX mode of operation. In: Roy, B.K., Meier, W. (eds.) FSE 2004. LNCS, vol. 3017, pp. 389–407. Springer, Berlin, Heidelberg (Feb 2004) 3

[12] Bellare, M., Shea, L.: Flexible password-based encryption: Securing cloud storage and provably resisting partitioning-oracle attacks. In: Rosulek, M. (ed.) CT-RSA 2023. LNCS, vol. 13871, pp. 594–621. Springer, Cham (Apr 2023) 3, 4

[13] Bhattacharjee, A., Bhaumik, R., Dhar, C.: Universal context commitment without ciphertext expansion. Cryptology ePrint Archive, Paper 2024/1382 (2024), https://eprint.iacr.org/2024/1382 3, 6, 29

[14] Chan, J., Rogaway, P.: On committing authenticated-encryption. In: Atluri, V., Di Pietro, R., Jensen, C.D., Meng, W. (eds.) ESORICS 2022, Part II. LNCS, vol. 13555, pp. 275–294. Springer, Cham (Sep 2022) 3, 4, 5, 12, 20

[15] Chen, S., Karadžić, V.: Committing authenticated encryption: Generic transforms with hash functions. In: Fehr, S., Fouque, P.A. (eds.) EUROCRYPT 2025, Part I. LNCS, vol. 15601, pp. 93–123. Springer, Cham (May 2025) 3, 5, 6

[16] Dobraunig, C., Eichlseder, M., Mendel, F., Schläffer, M.: Ascon v1.2: Lightweight authenticated encryption and hashing. Journal of Cryptology 34(3), 33 (Jul 2021) 6

[17] Dodis, Y., Grubbs, P., Ristenpart, T., Woodage, J.: Fast message franking: From invisible salamanders to encryptment. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018, Part I. LNCS, vol. 10991, pp. 155–186. Springer, Cham (Aug 2018) 3

[18] Dworkin, M.: Recommendation for block cipher modes of operation: Galois/Counter Mode (GCM) and GMAC. NIST Special Publication 800-38D (November 2007) 6, 29

[19] Dworkin, M.: The keyed-hash message authentication code (HMAC). FIPS PUB 198-1 (July 2008) 7, 14

[20] Dworkin, M.: Secure hash standard (SHS). FIPS PUB 180-4 (August 2015) 7, 14

[21] Dworkin, M.: PRE-DRAFT call for comments: NIST proposes to standardize a wider variant of AES. NIST Special Publication 800-197 (Initial Preliminary Draft) (December 2024), https://csrc.nist.gov/pubs/sp/800/197/iprd 7, 14

[22] Farshim, P., Orlandi, C., Roşie, R.: Security of symmetric primitives under incorrect usage of keys. IACR Trans. Symm. Cryptol. 2017(1), 449–473 (2017) 3

[23] Ferguson, N., Lucks, S., Schneier, B., Whiting, D., Bellare, M., Kohno, T., Callas, J., Walker, J.: The Skein hash function family. Submission to NIST (round 3) 7(7.5), 3 (2010) 14

[24] Gertner, Y., Herzberg, A.: Committing encryption and publicly-verifiable signcryption. Cryptology ePrint Archive, Report 2003/254 (2003), https://eprint.iacr.org/2003/254 3

[25] Grubbs, P., Lu, J., Ristenpart, T.: Message franking via committing authenticated encryption. In: Katz, J., Shacham, H. (eds.) CRYPTO 2017, Part III. LNCS, vol. 10403, pp. 66–97. Springer, Cham (Aug 2017) 3

[26] Gueron, S., Langley, A., Lindell, Y.: AES-GCM-SIV: Nonce misuse-resistant authenticated encryption. IETF Network Working Group, RFC 8452 (April 2019) 4, 8

[27] Hoang, V.T., Krovetz, T., Rogaway, P.: Robust authenticated-encryption AEZ and the problem that it solves. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015, Part I. LNCS, vol. 9056, pp. 15–44. Springer, Berlin, Heidelberg (Apr 2015) 4, 8, 13

[28] Hoang, V.T., Menda, S.: Robust AE with committing security. In: Chung, K.M., Sasaki, Y. (eds.) ASIACRYPT 2024, Part IX. LNCS, vol. 15492, pp. 312–342. Springer, Singapore (Dec 2024) 7, 12, 13

[29] Jean, J., Nikolic, I., Peyrin, T., Seurin, Y.: The Deoxys AEAD family. Journal of Cryptology 34(3), 31 (Jul 2021) 6

[30] Khairallah, M.: Revisiting leakage-resilient macs and succinctly-committing AEAD: More applications of pseudo-random injections. IACR Transactions on Symmetric Cryptology 2025(1), 211–239 (Mar 2025) 15

[31] Len, J., Grubbs, P., Ristenpart, T.: Partitioning oracle attacks. In: Bailey, M., Greenstadt, R. (eds.) USENIX Security 2021. pp. 195–212. USENIX Association (Aug 2021), https://www.usenix.org/conference/usenixsecurity21/presentation/len 3

[32] Liskov, M., Rivest, R.L., Wagner, D.: Tweakable block ciphers. Journal of Cryptology 24(3), 588–613 (Jul 2011) 7, 12

[33] Menda, S., Bellare, M., Hoang, V.T., Len, J., Ristenpart, T.: The OCH authenticated encryption scheme. In: Huang, C.Y., Chen, J.C., Shieh, S.P., Lie, D., Cortier, V. (eds.) ACM CCS 2025. pp. 2234–2248. ACM Press (Oct 2025) 4, 5, 6, 8

[34] Menda, S., Len, J., Grubbs, P., Ristenpart, T.: Context discovery and commitment attacks - how to break CCM, EAX, SIV, and more. In: Hazay, C., Stam, M. (eds.) EUROCRYPT 2023, Part IV. LNCS, vol. 14007, pp. 379–407. Springer, Cham (Apr 2023) 3, 12

[35] Namprempre, C., Rogaway, P., Shrimpton, T.: Reconsidering generic composition. In: Nguyen, P.Q., Oswald, E. (eds.) EUROCRYPT 2014. LNCS, vol. 8441, pp. 257–274. Springer, Berlin, Heidelberg (May 2014) 4, 11

[36] Namprempre, C., Rogaway, P., Shrimpton, T.: AE5 security notions: Definitions implicit in the CAESAR call. Cryptology ePrint Archive, Report 2013/242 (2013), https://eprint.iacr.org/2013/242 3, 4, 10

[37] Rogaway, P.: Authenticated-encryption with associated-data. In: Atluri, V. (ed.) ACM CCS 2002. pp. 98–107. ACM Press (Nov 2002) 3, 4, 5, 10, 23

[38] Rogaway, P., Bellare, M., Black, J., Krovetz, T.: OCB: A block-cipher mode of operation for efficient authenticated encryption. In: Reiter, M.K., Samarati, P. (eds.) ACM CCS 2001. pp. 196–205. ACM Press (Nov 2001) 3

[39] Rogaway, P., Shrimpton, T.: A provable-security treatment of the key-wrap problem. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 373–390. Springer, Berlin, Heidelberg (May / Jun 2006) 3, 4, 5, 6, 11, 13

[40] Thomson, M., Turner, S.: Using TLS to secure QUIC. IETF Network Working Group, RFC 9001 (May 2021) 4, 7

[41] Whiting, D., Housely, R., Ferguson, N.: Counter with CBC-MAC (CCM). IETF Network Working Group, RFC 3610 (September 2003) 3

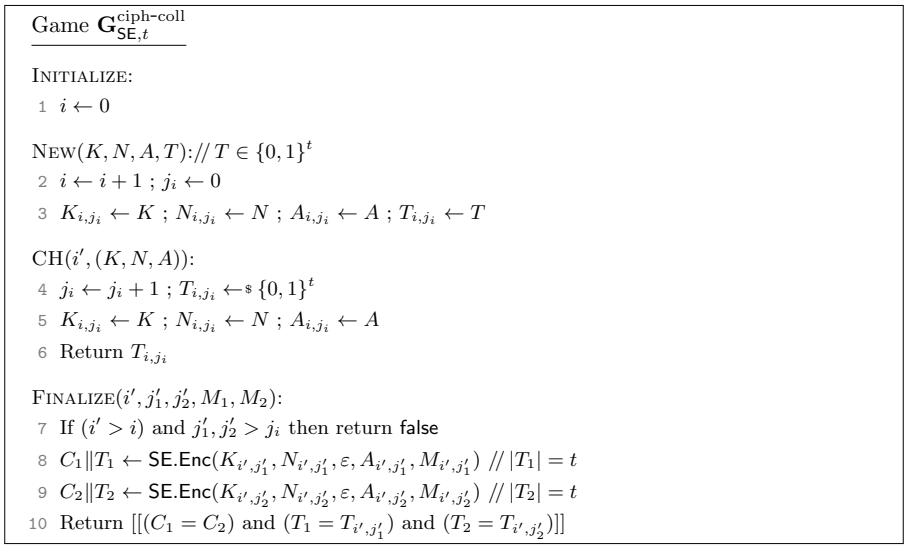


Figure 13: Game defining CIPH-COLL security of SE parameterized by a number $t > 0$.

A CIPH-COLL attack on GCM

Bhattacharjee, Bhaumik and Dhar (BBD24) [13] give a pair of transforms PACT and comPACT to add committing security to a base AE1 scheme SE, using as auxiliary tools a collision resistant hash function Hash and an ideal cipher E. For the transforms to provide committing security, however, BBD24 require that the base scheme SE satisfy a new and non-standard security notion called Ciphertext Collision Resistance (CIPH-COLL) that they define. BBD24 conjecture [13, Section 4.4] that any "well-designed" AE scheme should be CIPH-COLL-secure. Here we refute this conjecture. We give a simple attack that breaks the CIPH-COLL security of the widely standardized and used GCM AE1 scheme. This shows that the transforms of BBD24 are not guaranteed to work even for a very common choice of base scheme, namely GCM. As often with GCM, the attack exploits that GHASH is a polynomial hash and comes down to solving equations in the underlying finite field. We now proceed to the details.

CIPH-COLL-SECURITY. We first state the ciphertext collision definition for a symmetric encryption scheme SE from [13] as a game $\mathbf{G}^{\text{ciph-coll}}_{\mathsf{SE},t}$ in Figure 13. Here t is an integer parameter that, if the scheme is tag-based like GCM, will be the tag length. For an adversary $\mathcal{A}$ the advantage is defined as $\mathbf{Adv}^{\text{ciph-coll}}_{\mathsf{SE},t}(\mathcal{A}) = \Pr[\mathbf{G}^{\text{ciph-coll}}_{\mathsf{SE},t}(\mathcal{A})]$. Intuitively, the definitions asks that for any two partial contexts (K_1, N_1, A_1), (K_2, N_2, A_2) and t-bit string T_1 selected by the adversary, upon being given a random t-bit string T_2, it should be hard to find two messages M_1, M_2 such that $C\|T_1 = \mathsf{SE.Enc}(K_1, N_1, \varepsilon, A_1, M_1)$ and $C\|T_2 = \mathsf{SE.Enc}(K_2, N_2, \varepsilon, A_2, M_2)$ for some ciphertext prefix C. (Since this is for AE1, the nonce N is the public nonce and the secret nonce is ε.)

GCM MODE ENCRYPTION. In Figure 14 we give a simplified version of the encryption algorithm of GCM [18] based on block cipher $\mathsf{E} : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ and setting tag length $t = n$. We abstract out the pad created using counter mode in GCM using the function GCTR which is only based on the key K, initial counter $iv \leftarrow N\|0^{n-|N|-1}1$ and message length m and outputs a string of length m. The details of how GCTR computes the pad does not matter for our attack. There are different options for creating the initial counter in GCM based on length of N. We use the above since our adversary can always select a nonce of appropriate length that results in this. The

$\underline{\mathsf{GCM}[\mathsf{E},t].\textsc{Enc}(K,N,\varepsilon,A,M)}$:

1 $iv \leftarrow N\|0^{n-|N|-1}1$
2 $P \leftarrow \mathsf{GCTR}(K, iv, |M|)$
3 $C \leftarrow P \oplus M$
4 $H \leftarrow \mathsf{E}(K, 0^n)$
5 $X \leftarrow \mathsf{GHASH}(H, C, A)$
6 $T \leftarrow (X \oplus \mathsf{E}(K, iv))[1 \dots t]$
7 Return $C\|T$

Figure 14: The encryption algorithm for $\mathsf{GCM}[\mathsf{E}, t]$.

GHASH function is used to create the tag. It takes as input a key $H \leftarrow \mathsf{E}(K, 0^n)$, ciphertext prefix C and the header A. In our attack we put $A = \varepsilon$, so we describe GHASH in terms of H and C only. Note that, in the following, the operations are performed by representing the strings H, C_i (n-bit blocks of C) and length $c \leftarrow |C|$ (assumed to be a multiple of n) as appropriate field elements in $GF(2^n)$. Here, $+$ and $\cdot$ are addition and multiplication operations in $GF(2^n)$ and $m = c/n$. We have

$$\mathsf{GHASH}(H, C, \varepsilon) = \sum_{i=1}^{m} C_i \cdot H^{m-i+2} + c \cdot H \ .$$

Attack. We now describe our attack on CiphColl security of GCM mode of encryption based on block cipher $\mathsf{E}\colon \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ and tag length $t = n$. We give an adversary specifically for $n = 128$ so that, in GCM encryption, nonce length 96 results in the initial counter creation as described above. Adversary $\mathcal{A}$ is playing the $\mathbf{G}^{\text{ciph-coll}}_{\mathsf{GCM}[\mathsf{E},n]}$ game. In the following, operations in lines 6-11 are performed by first appropriately mapping the strings and numbers (represented as strings) to field elements in $GF(2^n)$. The output is then mapped back to binary representation when needed.

$\mathcal{A}()$:
1 $K_1, K_2 \leftarrow\!\!\$\ \{0,1\}^k$; $N_1 \leftarrow 0^{96}$; $N_2 \leftarrow 1^{96}$; $T_1 \leftarrow\!\!\$\ \{0,1\}^n$
2 $\textsc{New}(K_1, N_1, \varepsilon, T_1)$; $T_2 \leftarrow \mathrm{CH}(1, (K_2, N_2, \varepsilon))$
3 $H_1 \leftarrow \mathsf{E}(K_1, 0^n)$; $H_2 \leftarrow \mathsf{E}(K_2, 0^n)$
4 $Y_1 \leftarrow \mathsf{E}(K, N_1\|0^{31}1)$; $Y_2 \leftarrow \mathsf{E}(K, N_2\|0^{31}1)$
5 $X_1 \leftarrow T'_1 \oplus Y_1$; $X_2 \leftarrow T'_2 \oplus Y_2$
6 For unknown 2 block C, solve the following linear equations
7 $\quad C_1 \cdot H_1^3 + C_2 \cdot H_1^2 + \ell = X_1$
8 $\quad C_1 \cdot H_2^3 + C_2 \cdot H_2^2 + \ell = X_2$
9 where $\ell = 2n \cdot H$, to get
10 $C_1 \leftarrow (H_2^2 \cdot (X_1 + \ell) + H_1^2 \cdot (X_2 + \ell)) \cdot (H_1^3 \cdot H_2^2 + H_2^3 \cdot H_1^2)^{-1}$
11 $C_2 \leftarrow (H_2^3 \cdot (X_1 + \ell) + H_1^3 \cdot (X_2 + \ell)) \cdot (H_1^2 \cdot H_2^3 + H_2^2 \cdot H_1^3)^{-1}$
12 $P_1 \leftarrow \mathsf{GCTR}(K_1, N_1\|0^{31}1, 2n)$; $P_2 \leftarrow \mathsf{GCTR}(K_1, N_1\|0^{31}1, 2n)$
13 $M_1 \leftarrow P_1 \oplus C$; $M_2 \leftarrow P_2 \oplus C$
14 Return $(1, 0, 1, M_1, M_2)$

The idea for the attack is that the since the adversary knows the suffixes (tags) and nonces used for encryption, it can solve the GHASH equation to get the same ciphertext for both. And then get the corresponding messages which result in the obtained ciphertext and tags. We give an intuitive idea on why the adversary works. The attack is for $n = 128$. Let M_1, M_2 be the messages in the

output of $\mathcal{A}$. Notice that $M_1 = P_1 \oplus C$ and $M_2 = P_2 \oplus C$ for the C that adversary computes. Since P_1 and P_2 correspond to K_1, N_1 and K_2, N_2 respectively, the prefix part of ciphertexts for M_1 and M_2 will be C. Next in GCM encryption, $X_1 \leftarrow \mathsf{GHASH}(H_1, C, \varepsilon)$ and $X_2 \leftarrow \mathsf{GHASH}(H_2, C, \varepsilon)$ where $H_1 = \mathsf{E}(K_1, 0^n)$ and $H_2 = \mathsf{E}(K_2, 0^n)$. Therefore, the tags $T_1 \leftarrow (X_1 \oplus \mathsf{E}(K_1, N_1 \| 0^{31}1))[1 \dots t]$ and $T_2 \leftarrow (X_2 \oplus \mathsf{E}(K_2, N_2 \| 0^{31}1))[1 \dots t]$ will be as desired by the game since $\boldsymbol{X}_1, \boldsymbol{X}_2$ were created accordingly by the adversary. And so $\mathcal{A}$ always wins the game.