The proceedings version is in CRYPTO 2016. This is the full version.

The Multi-User Security of Authenticated Encryption: AES-GCM in TLS 1.3

Mihir Bellare[1] Björn Tackmann[2]

July 2, 2017

Abstract

We initiate the study of multi-user (mu) security of authenticated encryption (AE) schemes as a way to rigorously formulate, and answer, questions about the "randomized nonce" mechanism proposed for the use of the AE scheme GCM in TLS 1.3. We (1) Give definitions of mu ind (indistinguishability) and mu kr (key recovery) security for AE (2) Characterize the intent of nonce randomization as being improved mu security as a defense against mass surveillance (3) Cast the method as a (new) AE scheme RGCM (4) Analyze and compare the mu security of both GCM and RGCM in the model where the underlying block cipher is ideal, showing that the mu security of the latter is indeed superior in many practical contexts to that of the former, and (5) Propose an alternative AE scheme XGCM having the same efficiency as RGCM but better mu security and a more simple and modular design.

[1] Department of Computer Science & Engineering, University of California San Diego, 9500 Gilman Drive, La Jolla, California 92093, USA. Email: mihir@eng.ucsd.edu. URL: http://cseweb.ucsd.edu/~mihir/. Supported in part by NSF grants CNS-1526801 and CNS-1228890, ERC Project ERCC FP7/615074 and a gift from Microsoft.

[2] Department of Computer Science & Engineering, University of California San Diego, 9500 Gilman Drive, La Jolla, California 92093, USA. Email: btackmann@eng.ucsd.edu. URL: http://cseweb.ucsd.edu/~btackmann. Supported in part by the Swiss National Science Foundation (SNF) via Fellowship No. P2EZP2_155566 and NSF grant CNS-1228890.

Contents

1 Introduction

Traditionally, security definitions were single-user, meaning there was a single target key. Consideration of the multi-user setting began with public-key encryption [3]. In this setting, there are many users, each with their own key, and the target is to violate security under *some* key. This is, first, simply more realistic, reflecting real usage, but is now even more relevant from the mass-surveillance perspective. This paper initiates a study of the multi-user security of authenticated encryption. Our motivation comes from TLS 1.3.

AE. The form of authenticated encryption (AE) we consider is nonce-based [28]. The encryption algorithm $\mathsf{AE.Enc}$ takes key K, nonce N, message M and header H to deterministically return a ciphertext $C \leftarrow \mathsf{AE.Enc}(K, N, M, H)$. The requirement formalized in [28] is to provide privacy of M, and authenticity of both M and H, as long as a nonce is not re-used. The formalization refers to only one target key, meaning is in the single user (su) setting.

There are many AE schemes (provably) meeting this security requirement. One simple way to obtain them is via generic composition of privacy-only encryption schemes with MACs [5, 26]. There are also dedicated schemes such as OCB [31, 29, 22], CCM [11] and GCM [24, 12]. The last, with AES, is used in TLS 1.3.

Multi-User Security of AE. We formalize multi-user (mu) security of an authenticated-encryption scheme AE. The game picks an adversary-determined number u of independent target keys $K[1], \ldots, K[u]$. The adversary gets an encryption oracle that takes an index $i \in [1..u]$, a message, nonce and header, and returns either an encryption of these under $K[i]$ or a random string of the same length. It also gets a verification oracle that takes i, a ciphertext, nonce and header, and indicates whether or not decryption is valid. Security is required as long as the adversary does not re-use a nonce *for a particular user*. That is, it is fine to obtain encryptions under the same nonce for different keys, just not under the same key. When $u = 1$, we get a definition equivalent to (but formulated slightly differently from) the (single-user) definition of [28].

Besides this usual goal (which we call indistinguishability), we also formalize a mu key-recovery goal. Again the game picks target keys $K[1], \ldots, K[u]$ and gives the adversary an encryption oracle. This time time the latter is always true, meaning it takes an index $i \in [1..u]$, a message, nonce and header, and returns an encryption of these under $K[i]$. The adversary also gets a verification oracle, and, to win, must find one of the target keys. A key-recovery attack is much more damaging than a distinguishing attack, and is the threat of greatest concern to practitioners. Key-recovery security is usually dismissed by theoreticians as being implied by indistinguishability, but this view misses the fact that the quantitative security of a scheme, in terms of bounds on adversary advantage, can be very different for the two metrics, making it worthwhile to consider key-recovery security separately and additionally.

We give our definitions in the ideal-cipher model. (Standard-model definitions follow because this is just the special case where scheme algorithms and adversaries make no queries to the ideal cipher.) For all the schemes we consider, the assumption that the underlying block cipher is a PRP suffices to prove security. The reason we use the ideal-cipher model is that adversary queries to the ideal cipher give a clear and rigorous way to measure the offline computation being performed in an attack. Also in some cases we get better bounds.

Multi-user security is not qualitatively different from single-user security. A hybrid argument shows that the latter implies the former. But the two could be quantitatively quite different, and this has important practical implications. In the hybrid reduction, there is a loss of a factor u in adversary advantage. Thus, the mu advantage of an adversary could be as much as u times its su advantage. This is the worst case. But it could be a lot less, degrading much more slowly with u.

This would be better.

AE IN TLS 1.3. As the protocol underlying `https`, TLS is the basis for secure communication on the Internet, used millions of times a day. The existing versions up to TLS 1.2 have however been subject to many attacks. The effort to create a new and (hopefully) better version, TLS 1.3, is currently underway. TLS (of whatever version) begins with a *handshake*. This is an authenticated key exchange that establishes a shared session key, called the traffic secret, between client and server. This step will not be our concern. After the handshake, data is authenticated and encrypted within the so-called *record layer*, using an authenticated encryption scheme AE that is keyed by a key K derived from the traffic secret. The currently proposed choice of AE is AES-GCM.

The most natural way to use AE in the record layer is directly, meaning the data message M is simply encrypted via $C \leftarrow \mathsf{AE.Enc}(K, N, M, H)$, where N is a nonce (in TLS 1.3 this is a sequence number that is known to the receiver) and H is the header. This is not what TLS 1.3 proposes. Instead, they randomize the nonce, computing $C \leftarrow \mathsf{AE.Enc}(K, N \oplus L, M, H)$, where the randomizer L is also derived from the traffic secret. (It is thus known to the receiver, enabling decryption.) Why do this? Brian Smith gave the following motivation on the TLS 1.3 mailing list [33]:

> ... massively parallel attacks on many keys at once seem like the most promising way to break AES-128. It seems bad to have popular endpoints encrypting the same plaintext block with the same nonce with different keys. That seems like exactly the recipe for making such attacks succeed. It seems like it would be better, instead, to require that the initial nonces to be calculated from the key block established during key agreement ... This ... should prevent any such massively-parallel attack from working.

In this paper, we aim to understand and formalize the threat alluded to here, and then assess to what extent one can prove that nonce-randomization guarantees security. In particular, we suggest that the formal cryptographic goal underlying nonce randomization and Smith's comment is improved multi-user security. In our model, the "massively parallel attack" is a key-search attack that finds the GCM key of some user out of u target users—here we are referring to the basic GCM scheme, in the absence of nonce randomization—in time $2^\kappa/u$ where κ is the key length of the underlying block cipher, $\kappa = 128$ for AES. The attack picks some N, M, H and for each $i \in [1..u]$ obtains from its encryption oracle the encryption C_i of these quantities under K_i. Now, it goes through all possible κ-bit keys L, for each computing $C_L \leftarrow \mathsf{AE.Enc}(L, N, M, H)$, and returning L if $C_L = C_i$ for some i. Note that the attack needs a single computation of $\mathsf{AE.Enc}$ for each L, not one per user, which is why the running time is $2^\kappa/u$. Given NSA computing capabilities, the fear of the TLS 1.3 designers is that this attack may be feasible for them for large u, and thus a mass-surveillance threat. Nonce randomization is a candidate way to circumvent the attack. The question this raises is whether nonce randomization works. To answer this in a rigorous way, we abstract out a (new) AE scheme and then use our definitions of mu security.

RGCM. In TLS 1.3, nonce randomization is viewed as a way to use GCM in the record layer. We take a different perspective. We view the method as defining a new AE scheme that we call RGCM. In this scheme, the randomizer is part of the key. This view is appropriate because the randomizer was derived from the traffic secret just like the base key, and has the security necessary to be used as a key, and the randomizer is also static across the session, just like the base key. While GCM has a key whose length is the key length κ of the underlying block cipher ($\kappa = 128$ for AES), RGCM has a key of length $\kappa + \nu$, where ν is the length of the randomizer ($\nu = 96$ for GCM in TLS 1.3). Nonces are assumed to also have length ν so that xoring the nonce with the randomizer makes sense.

Scheme	Key length	kr security bound
GCM/CAU	κ	$\frac{u(p+1)}{2^\kappa}$
RGCM/RCAU	$\kappa+\nu$	$\frac{u^2+8p_i}{2^\kappa}+\frac{upq_e}{2^{\kappa+\nu-4}}+\frac{upm}{2^{\kappa+\lambda-6}}+\frac{(um+p)(um)^3}{2^{-2\lambda}}$
XGCM/XCAU	$\kappa+\lambda$	$\frac{u}{2^\kappa}+\frac{upm}{2^{\kappa+\lambda+1}}$

Table 1: Comparison between the schemes, where λ is the block length of the block cipher, ν is the nonce length, u is the number of users, p is the number of adversary block cipher evaluations (p_i the number of inversion queries), m is the number of bits encrypted per user in total, i.e., the sum of the lengths of all messages in the queries, and q_e is the number of messages encrypted per user.

Results. With this perspective, we are looking at two AE schemes, GCM and RGCM. We can now divorce ourselves of TLS details and analyze them as AE schemes to determine the quantitative mu security of both. The number p of adversary queries to the ideal cipher is the central parameter, capturing the offline computational effort of the adversary. As before u is the number of users, and we let m denote the total number of bits encrypted, meaning the sum of the lengths of all messages in queries.

Let us first discuss mu security under key recovery. Roughly, we show that key recovery for GCM needs $p = 2^\kappa/u$ while for RGCM (here only in the case where the adversary does not query its verification oracle), it needs $p_i \approx 2^\kappa + 2^{\kappa+\nu}/um$. The first term does not scale with the number of honest users, and as we expect m to be quite a bit less than 2^ν—in the current schemes, $\nu = 96$—the effort to recover a key is significantly higher for RGCM than for GCM. This says that nonce randomization works, meaning it does increase mu security as targeted by the TLS 1.3 designers, at least for key recovery.

For mu-ind security, our bounds are complex, and interesting terms get swamped by collision terms. We stress that the bounds here may not be tight, so the picture we are seeing could reflect limitations of our analysis techniques rather than the inherent security of the schemes. Obtaining better (and ideally tight) bounds is an interesting open question.

XGCM. Even if under some metrics superior to GCM, RGCM performs considerably worse than expected from an AE with key length $\kappa+\nu$, and the natural question is, why not use some standard scheme or construction paradigm rather than "roll your own" with RGCM? The most obvious choice is AES256-GCM. Our analysis of GCM shows that AES256-GCM has good enough mu security, simply due to the larger key size. However, AES256-GCM is slower than AES-RGCM, and a scheme using AES itself would be preferable. We suggest and analyze XGCM, derived simply as GCM with the block cipher E: $\{0,1\}^\kappa \times \{0,1\}^\lambda \to \{0,1\}^\lambda$ replaced by EX: $\{0,1\}^{\kappa+\lambda} \times \{0,1\}^\lambda \to \{0,1\}^\lambda$, defined by $\mathrm{EX}(K\|L, X) = L \oplus \mathrm{E}(K, L \oplus X)$. This transform of a block cipher uses the Even-Mansour technique [13]. It was suggested by Rivest as a key-extension method for DES and first analyzed by Kilian and Rogaway [19]. Our analysis implies that, with AES parameters, the mu security of XGCM is better than that of RGCM. Its performance is however essentially the same as that of GCM or RGCM. While it would be a viable alternative for AES-RGCM in TLS 1.3, it does require non-black-box changes to the implementation of AES-GCM, whereas for AES-RGCM the change is only the randomization of the nonce input. A comparison between all three discussed schemes is given in Table 1.

RELATED WORK. GCM was proposed by McGrew and Viega (MV) [24] and standardized by NIST as [12]. MV [24] prove single-user security assuming PRP-security of the underlying block cipher. While the original scheme allows variable-length nonces [24], IOM [18] showed that the security proof of MV was flawed in this case and the claimed security bounds did not hold. They provide a corrected proof, which was later improved by NOMI [27]. In this paper we only consider fixed-length nonces. We prove security in the mu setting in the ideal cipher model.

Key-recovery security of symmetric encryption schemes was defined in [30] for the single-user, privacy-only setting. We extend their definition to the mu, authenticated encryption setting.

BMMRT [1] and FGMP [14] analyze the record layer of TLS 1.3 relative to the goal of providing a secure channel, under an appropriate formalization of the latter. These works assume that AES-GCM is a secure AE scheme. Our work is not attempting to analyze the record layer. It is analyzing the security of GCM and RGCM as stand-alone AE schemes, with emphasis on their mu security.

We are seeing increased interest in multi-user security, further reflected in this paper. BCK [4] considered mu security for PRFs as an intermediate step in the analysis of the cascade construction. Multi-user security of PRFs and PRPs (block ciphers) has been further considered in [25, 34, 2]. The first work that highlighted mu security as a goal and targeted quantitative security improvements seems to have been BBM [3], the primitive here being public-key encryption. Multi-user security for signatures was considered by GMS [16] and has been the subject of renewed interest in [20, 8]. Further works involving multi-user security include [9, 10, 17], and, in the cryptanalytic context, [15].

2 Preliminaries

We let ε denote the empty string. If Z is a string then $|Z|$ denotes its length and $Z[1..i]$ denotes bits 1 through i of Z. If X is a finite set, we let $x \xleftarrow{\$} X$ denote picking an element of X uniformly at random and assigning it to x. Algorithms may be randomized unless otherwise indicated. Running time is worst case. If A is an algorithm, we let $y \leftarrow A(x_1, \ldots; r)$ denote running A with random coins r on inputs $x_1, \ldots$ and assigning the output to y. We let $y \xleftarrow{\$} A(x_1, \ldots)$ be the result of picking r at random and letting $y \leftarrow A(x_1, \ldots; r)$. We let $[A(x_1, \ldots)]$ denote the set of all possible outputs of A when invoked with inputs $x_1, \ldots$.

We use the code-based game-playing framework of BR [6]. (See Fig. 1 for an example.) By $\Pr[\mathrm{G}]$ we denote the probability that the execution of game G results in the game returning true. In games, integer variables, set variables and boolean variables are assumed initialized, respectively, to 0, the empty set, and false.

A family of functions $\mathsf{F}\colon \mathsf{F.Keys} \times \mathsf{F.Dom} \to \mathsf{F.Rng}$ is a two-argument function that takes a key K in the key space $\mathsf{F.Keys}$, an input x in the domain $\mathsf{F.Dom}$ and returns an output $\mathsf{F}(K, x)$ in the range $\mathsf{F.Rng}$. In the ROM, F takes an oracle RO. We say F has key length $\mathsf{F.kl}$ if $\mathsf{F.Keys} = \{0,1\}^{\mathsf{F.kl}}$; output length $\mathsf{F.ol}$ if $\mathsf{F.Rng} = \{0,1\}^{\mathsf{F.ol}}$; and input length $\mathsf{F.il}$ if $\mathsf{F.Dom} = \{0,1\}^{\mathsf{F.il}}$.

We say that $\mathsf{F}\colon \{0,1\}^{\mathsf{F.kl}} \times \{0,1\}^{\mathsf{F.il}} \to \{0,1\}^{\mathsf{F.ol}}$ is a *block cipher* if $\mathsf{F.il} = \mathsf{F.ol}$ and $\mathsf{F}(K, \cdot)\colon \{0,1\}^{\mathsf{F.il}} \to \{0,1\}^{\mathsf{F.ol}}$ is a permutation for each K in $\{0,1\}^{\mathsf{F.kl}}$. We denote by $\mathsf{F}^{-1}(K, \cdot)$ the inverse of $\mathsf{F}(K, \cdot)$.

Let $\mathsf{H}\colon \mathsf{H.Keys} \times (\{0,1\}^* \times \{0,1\}^*) \to \{0,1\}^{\mathsf{H.ol}}$ be a family of functions with domain $\mathsf{H.Dom} = \{0,1\}^* \times \{0,1\}^*$. Let $\epsilon\colon \mathbb{N} \times \mathbb{N} \to [0,1]$ be a function. Somewhat extending [21], we say that H is *ϵ-almost XOR-universal* if for all distinct $(M_1, H_1), (M_2, H_2) \in \mathsf{H.Dom}$ and all $s \in \{0,1\}^{\mathsf{H.ol}}$, we have

$$\begin{aligned}\Pr[\mathsf{H}(hk, (M_1, H_1)) \oplus \mathsf{H}(hk, (M_2, H_2)) = s \ :\ hk \xleftarrow{\$} \mathsf{H.Keys}] \\ \leq \epsilon(\max(|M_1|, |M_2|), \max(|H_1|, |H_2|))\ .\end{aligned}$$

Game $\mathbf{G}^{\text{mu-ind}}_{\mathsf{AE}}(A)$	$\underline{\text{VF}(i,N,C,H)}$
$b \leftarrow\!\!\$\ \{0,1\}$; $b' \leftarrow\!\!\$\ A^{\text{NEW},\text{ENC},\text{VF},\text{E},\text{E}^{-1}}$	If not $(1 \le i \le v)$ then return $\perp$
Return $(b' = b)$	If $((i,N,C,H) \in W)$ then return true
$\underline{\text{NEW}()}$	If $(b = 0)$ then return false
$v \leftarrow v + 1$; $K[v] \leftarrow\!\!\$\ \{0,1\}^{\mathsf{AE.kl}}$	$M \leftarrow \mathsf{AE.Dec}^{\text{E},\text{E}^{-1}}(K[i],N,C,H)$
$\underline{\text{ENC}(i,N,M,H)}$	Return $(M \neq \perp)$
If not $(1 \le i \le v)$ then return $\perp$	$\underline{\text{E}(L,x)}$
If $((i,N) \in V)$ then return $\perp$	If $T[L,x] = \perp$ then
$C_1 \leftarrow \mathsf{AE.Enc}^{\text{E},\text{E}^{-1}}(K[i],N,M,H)$	$\quad T[L,x] \leftarrow\!\!\$\ \overline{\text{im}\, T[L,\cdot]}$
$C_0 \leftarrow\!\!\$\ \{0,1\}^{\mathsf{AE.cl}(\lvert M\rvert)}$	Return $T[L,x]$
$V \leftarrow V \cup \{(v,N)\}$; $W \leftarrow W \cup \{(i,N,C_b,H)\}$	$\underline{\text{E}^{-1}(L,y)}$
Return C_b	If $T^{-1}[L,y] = \perp$ then
	$\quad T^{-1}[L,y] \leftarrow\!\!\$\ \overline{\text{im}\, T^{-1}[L,\cdot]}$
	Return $T^{-1}[L,y]$

Figure 1: Game defining multi-user indistinguishability security of symmetric encryption scheme AE in the ideal-cipher model.

3 Multi-User Security of Symmetric Encryption

We consider symmetric encryption in a multi-user setting. We give two definitions of security. The first, an indistinguishability-style definition, extends Rogaway's single-user definition [28] to the multi-user setting, and represents a very strong requirement. We also define security against key recovery, representing the goal the attacker would most like to achieve and the most common target of cryptanalysis. We will see that the security bounds for these notions can differ. Since our analyses will be in the ideal-cipher model, the definitions are given directly in that model.

SYNTAX. A symmetric encryption scheme AE specifies a deterministic encryption algorithm $\mathsf{AE.Enc} : \{0,1\}^{\mathsf{AE.kl}} \times \mathsf{AE.NS} \times \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^*$ that takes a key $K \in \{0,1\}^{\mathsf{AE.kl}}$, a nonce $N \in \mathsf{AE.NS}$, a message $M \in \{0,1\}^*$ and a header $H \in \{0,1\}^*$ to return a ciphertext $C \leftarrow \mathsf{AE.Enc}^{\text{E},\text{E}^{-1}}(K,N,M,H) \in \{0,1\}^{\mathsf{AE.cl}(\lvert M\rvert)}$. Here $\mathsf{AE.kl} \in \mathbb{N}$ is the key length of the scheme, $\mathsf{AE.NS}$ is the nonce space and $\mathsf{AE.cl}\colon \mathbb{N} \to \mathbb{N}$ is the ciphertext length function. The oracles represent a cipher $\text{E}\colon \{0,1\}^{\mathsf{AE.ckl}} \times \{0,1\}^{\mathsf{AE.bl}} \to \{0,1\}^{\mathsf{AE.bl}}$ and its inverse E^{-1}. In the security games this cipher will be chosen at random, meaning be ideal. We view the key length $\mathsf{AE.ckl}$ and block length $\mathsf{AE.bl}$ of the cipher as further parameters of AE itself. Also specified is a deterministic decryption algorithm $\mathsf{AE.Dec}\colon \{0,1\}^{\mathsf{AE.kl}} \times \mathsf{AE.NS} \times \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^* \cup \{\perp\}$ that takes K,N,C,H and returns $M \leftarrow \mathsf{AE.Dec}^{\text{E},\text{E}^{-1}}(K,N,C,H) \in \{0,1\}^* \cup \{\perp\}$. Correctness requires that $\mathsf{AE.Dec}(K,N,\mathsf{AE.Enc}(K,N,M,H),H) = M$ for all $M,H \in \{0,1\}^*$, all $N \in \mathsf{AE.NS}$ and all $K \in \{0,1\}^{\mathsf{AE.kl}}$.

INDISTINGUISHABILITY SECURITY. We extend Rogaway's definition of indistinguishability security for authenticated encryption [28], which is in the single-user setting, to the multi-user setting. The formalization is based on game $\mathbf{G}^{\text{mu-ind}}_{\mathsf{AE}}(A)$ of Fig. 1, associated to encryption scheme AE and adversary A. The game initially samples a random challenge bit b, with $b = 1$ indicating it is in "real" mode and $b = 0$ that it is in "ideal" mode. As per our conventions noted in Section 2, the sets V, W are assumed initialized to the empty set, and the integer v is assumed initialized to 0. Now the adversary A has access to an oracle NEW that creates new user instances. A also has access

to an encryption oracle ENC that takes a user instance identifier i, a nonce $N \in \mathsf{AE.NS}$, a message M, and a header H. The oracle either returns a uniformly random bit string of length $\mathsf{AE.cl}$ that depends only on the length of M (for $b = 0$), or an encryption under $\mathsf{AE.Enc}$ using the key of user i (for $b = 1$). The oracle checks that A does not re-use nonces for a user instance, and that it is invoked only for user instances that exist. Analogously, there is a verification oracle VF that takes user instance i, nonce $N \in \mathsf{AE.NS}$, ciphertext C, and header H. Oracle VF always accepts ciphertexts generated by ENC for the same i, N, and H, rejects all other ciphertexts for $b = 0$, and uses the decryption algorithm $\mathsf{AE.Dec}$ to check the validity of the ciphertext for $b = 1$. As a last step, the adversary outputs a bit b' that can be viewed as a guess for b. The advantage of adversary A in breaking the mu-ind security of AE is defined as $\mathsf{Adv}^{\text{mu-ind}}_{\mathsf{AE}}(A) = 2\Pr[\mathbf{G}^{\text{mu-ind}}_{\mathsf{AE}}(A)] - 1$.

The ideal-cipher oracles E and E^{-1} are given to the adversary, the encryption algorithm and the decryption algorithm, where the inputs are $L \in \{0,1\}^{\mathsf{AE.ckl}}$ and $x, y \in \{0,1\}^{\mathsf{AE.bl}}$. The oracles are defined using lazy sampling. The description of game $\mathbf{G}^{\text{mu-ind}}_{\mathsf{AE}}$ in Fig. 1 uses some notation that we introduce here and use also elsewhere. First of all, $T[\cdot,\cdot]$ describes a map $\{0,1\}^* \times \{0,1\}^* \to \{0,1\}^*$ that is initially $\perp$ everywhere, with new values defined during the game. By $\operatorname{im} T[\cdot,\cdot]$ we denote the set $\{z \in \{0,1\}^* : \exists x, y \in \{0,1\}^* \text{ with } T[x,y] = z\}$ and by $\operatorname{supp} T[\cdot,\cdot]$ the set $\{(x,y) \in \{0,1\}^* \times \{0,1\}^* : T[x,y] \neq \perp\}$. Both terms are also used in the obvious sense in settings where one of the inputs is fixed. (In Fig. 1, this input is L.) Since $T[L,\cdot]$ is a permutation for any L, we implicitly use that setting a value $T[L,x] \leftarrow y$ also defines $T^{-1}[L,y] = x$, and vice versa. Finally, for a subset $A \subset B$, the notation $\overline{A}$ refers to the complement $B \setminus A$ in B. We use this notation in places when the superset B is clear from the context. (In Fig. 1, the set B is $\{0,1\}^{\mathsf{AE.bl}}$.)

Definitions of mu security for authenticated encryption in the standard model are obtained as a special case, namely by restricting attention to schemes and adversaries that do not make use of the E and E^{-1} oracles.

One can further strengthen the security of the above ind definition by considering *nonce-misuse resistance* as defined by Rogaway and Shrimpton [32]. This requires changing the condition $(i,N) \in U$ in oracle ENC to only prevent queries where nonce *and* message (or even nonce, message, and header) are repeated. We do not use such a stronger definition in this work because GCM does not achieve it.

Rogaway's definition of indistinguishability security for authenticated encryption (in the su setting) [28] gives the adversary a decryption oracle, while we give it a verification oracle. The latter is simpler and our definition can be shown equivalent to one with a decryption oracle by the technique of BN [5].

KEY-RECOVERY SECURITY. The qualitatively weaker requirement of key-recovery security can sometimes be established with better bounds than ind, which is of practical importance since violating key recovery is much more damaging that violating ind. The formalization is based on game $\mathbf{G}^{\text{mu-kr}}_{\mathsf{AE}}(A)$ of Fig. 2, associated to encryption scheme AE and adversary A. The goal of the adversary A is simply to output the key of any honest user. It again has access to oracles NEW, ENC, VF, E, and E^{-1}. Oracles ENC and VF are defined to always return the values as determined by the scheme AE. Adversary A wins if it outputs any one of the keys that was generated using the NEW oracle. The advantage of A in breaking the mu-kr security of AE is defined as $\mathsf{Adv}^{\text{mu-kr}}_{\mathsf{AE}}(A) = \Pr[\mathbf{G}^{\text{mu-kr}}_{\mathsf{AE}}(A)]$.

4 The Schemes

We present a symmetric encryption scheme we call CAU, for Counter-Mode with a AXU hash function. GCM is a special case. This allows us to divorce our results and analyses from some

Game $\mathbf{G}^{\text{mu-kr}}_{\mathsf{AE}}(A)$	$\textsc{Vf}(i, N, C, H)$
$\bar{K} \leftarrow\!\!\$\ A^{\textsc{New},\textsc{Enc},\textsc{Vf},\mathrm{E},\mathrm{E}^{-1}}$	If not $(1 \le i \le v)$ then return $\bot$
Return $(\bar{K} \in \{K[1], \ldots, K[v]\})$	$M \leftarrow \mathsf{AE.Dec}^{\mathrm{E},\mathrm{E}^{-1}}(K[i], N, C, H)$
$\textsc{New}()$	Return $(M \neq \bot)$
$v \leftarrow v + 1$; $K[v] \leftarrow\!\!\$\ \{0,1\}^{\mathsf{AE.kl}}$	$\mathrm{E}(L, x)$
$\textsc{Enc}(i, N, M, H)$	If $T[L, x] = \bot$ then
If not $(1 \le i \le v)$ then return $\bot$	$\quad T[L, x] \leftarrow\!\!\$\ \overline{\mathrm{im}\, T[L, \cdot]}$
If $((i, N) \in V)$ then return $\bot$	Return $T[L, x]$
$C \leftarrow \mathsf{AE.Enc}^{\mathrm{E},\mathrm{E}^{-1}}(K[i], N, M, H)$	$\mathrm{E}^{-1}(L, y)$
$V \leftarrow V \cup \{(i, N)\}$	If $T^{-1}[L, y] = \bot$ then
Return C	$\quad T^{-1}[L, y] \leftarrow\!\!\$\ \overline{\mathrm{im}\, T^{-1}[L, \cdot]}$
	Return $T^{-1}[L, y]$

Figure 2: Game defining multi-user key-recovery security of symmetric encryption scheme AE in the ideal-cipher model.

$\mathsf{CAU.Enc}^{\mathrm{E},\mathrm{E}^{-1}}(K, N, M, H)$	$\mathsf{CAU.Dec}^{\mathrm{E},\mathrm{E}^{-1}}(K, N, T \| C, H)$
$\ell \leftarrow \lceil\, \lvert M\rvert/\lambda\, \rceil$	$\ell \leftarrow \lceil\, \lvert M\rvert/\lambda\, \rceil - 1$
$M_1 \| \ldots \| M_\ell \leftarrow M$ // block length λ	$C_1 \| \ldots \| C_\ell \leftarrow C$ // block length λ
$r \leftarrow \lvert M_\ell\rvert$ // last block length	$r \leftarrow \lvert C_\ell\rvert$ // last block length
$G \leftarrow \mathrm{E}(K, 0^\lambda)$; $Y \leftarrow N \| \langle 1 \rangle$	$G \leftarrow \mathrm{E}(K, 0^\lambda)$; $Y \leftarrow N \| \langle 1 \rangle$
For $i = 1$ to $\ell - 1$	$T' \leftarrow \mathsf{H}(G, H, C) \oplus \mathrm{E}(K, Y + 0)$
$\quad C_i \leftarrow M_i \oplus \mathrm{E}(K, Y + i)$	If $T \neq T'$ then return $\bot$
$C_\ell \leftarrow M_\ell \oplus \mathrm{msb}_r(\mathrm{E}(K, Y + \ell))$	For $i = 1$ to $\ell - 1$
$C \leftarrow C_1 \| \ldots \| C_\ell$	$\quad M_i \leftarrow C_i \oplus \mathrm{E}(K, Y + i)$
$T \leftarrow \mathsf{H}(G, H, C) \oplus \mathrm{E}(K, Y + 0)$	$M_\ell \leftarrow C_\ell \oplus \mathrm{msb}_r(\mathrm{E}(K, Y + \ell))$
Return $T \| C$	Return $M_1 \| \ldots \| M_\ell$

Figure 3: Encryption scheme $\mathsf{CAU} = \mathbf{CAU}[\mathsf{H}, \kappa, \lambda, \nu]$. **Left:** Encryption algorithm $\mathsf{CAU.Enc}$. **Right:** Decryption algorithm $\mathsf{CAU.Dec}$.

details of GCM (namely, the particular, polynomial-evaluation based hash function) making them both simpler and more general.

The TLS Working Group introduced a specific usage mode of GCM in recent draft versions of TLS 1.3 in which material, obtained in the handshake key-derivation phase, is used to mask the nonce. We take a different perspective and view this as a new symmetric encryption scheme whose generalized version we specify here as RCAU. Finally we specify XCAU, our own variant that better achieves the same goals.

CAU. Let $\kappa, \lambda, \nu \ge 1$ be integers such that $\nu \le \lambda - 2$, where κ is referred to as the *cipher key length*, λ as the *block length* and ν as the *nonce length*. Let H: $\{0,1\}^\lambda \times (\{0,1\}^* \times \{0,1\}^*) \to \{0,1\}^\lambda$ be an ϵ-XOR universal hash function. We associate to these the symmetric encryption scheme $\mathsf{CAU} = \mathbf{CAU}[\mathsf{H}, \kappa, \lambda, \nu]$—here **CAU** is a transform taking $\mathsf{H}, \kappa, \lambda, \nu$ and returning a symmetric encryption scheme that we are denoting CAU—whose encryption and decryption algorithms are

$\underline{\mathsf{RCAU.Enc}^{\mathrm{E},\mathrm{E}^{-1}}(K\|L,N,M,H)}$	$\underline{\mathsf{RCAU.Dec}^{\mathrm{E},\mathrm{E}^{-1}}(K\|L,N,T\|C,H)}$
$\ell \leftarrow \lceil\, \|M\|/\lambda \,\rceil$	$\ell \leftarrow \lceil\, \|M\|/\lambda \,\rceil - 1$
$M_1\| \ldots \|M_\ell \leftarrow M$ // block length λ	$C_1\| \ldots \|C_\ell \leftarrow C$ // block length λ
$r \leftarrow \|M_\ell\|$ // last block length	$r \leftarrow \|C_\ell\|$ // last block length
$G \leftarrow \mathrm{E}(K, 0^\lambda)$; $Y \leftarrow (N \oplus L)\|\langle 1\rangle$	$G \leftarrow \mathrm{E}(K, 0^\lambda)$; $Y \leftarrow (N \oplus L)\|\langle 1\rangle$
For $i = 1$ to $\ell - 1$	$T' \leftarrow \mathsf{H}(G, H, C) \oplus \mathrm{E}(K, Y + 0)$
$\quad C_i \leftarrow M_i \oplus \mathrm{E}(K, Y + i)$	If $T \neq T'$ then return $\perp$
$C_\ell \leftarrow M_\ell \oplus \mathrm{msb}_r(\mathrm{E}(K, Y + \ell))$	For $i = 1$ to $\ell - 1$
$C \leftarrow C_1\| \ldots \|C_\ell$	$\quad M_i \leftarrow C_i \oplus \mathrm{E}(K, Y + i)$
$T \leftarrow \mathsf{H}(G, H, C) \oplus \mathrm{E}(K, Y + 0)$	$M_\ell \leftarrow C_\ell \oplus \mathrm{msb}_r(\mathrm{E}(K, Y + \ell))$
Return $T\|C$	Return $M_1\| \ldots \|M_\ell$

Figure 4: Encryption scheme $\mathsf{RCAU} = \mathbf{RCAU}[\mathsf{H}, \kappa, \lambda, \nu]$. **Left:** Encryption algorithm RCAU.Enc. **Right:** Decryption algorithm RCAU.Dec.

---

specified in Fig. 3. The scheme has key length $\mathsf{CAU.kl} = \kappa$, cipher key length $\mathsf{CAU.ckl} = \kappa$ and block length $\mathsf{CAU.bl} = \lambda$. It has nonce space $\mathsf{CAU.NS} = \{0,1\}^\nu$ and ciphertext length function $\mathsf{CAU.cl}(\cdot)$ defined by $\mathsf{CAU.cl}(m) = m + \lambda$. Explanations follow.

The algorithms CAU.Enc and CAU.Dec are given access to oracles that represent a cipher $\mathrm{E}\colon \{0,1\}^\kappa \times \{0,1\}^\lambda \to \{0,1\}^\lambda$ and its inverse E^{-1}. In the security games the cipher will be chosen at random, meaning be ideal. In practice, it will be instantiated by a block cipher, usually AES.

CAU is an encrypt-then-mac scheme [5]. Encryption is counter-mode of the block cipher. The MAC is a Carter-Wegman MAC based on the AXU function family H. Some optimizations are performed over and above generic encrypt-then-mac to use the same key for both parts. The name stands for "Counter Almost Universal."

In the description of Fig. 3, the plaintext M is first partitioned into $\ell = \lceil |M|/\lambda \rceil$ plaintext blocks $M_1, \ldots, M_\ell$. The first $\ell - 1$ blocks have length λ. The final block M_ℓ has length $1 \le r \le \lambda$. The value G defined as $\mathrm{E}(K, 0^\lambda)$ is later used as a key for the hash function H. The loop then computes the counter-mode encryption. Here and in the rest of the paper we use the following notation. If Z is a λ bit string and $j \ge 0$ is an integer then we let

$$Z + j = Z[1..\nu]\|\langle 1 + j\rangle \tag{1}$$

where $\langle 1 + j\rangle$ is the representation of the integer $(1 + j) \bmod 2^{\lambda-\nu}$ as a $(\lambda - \nu)$-bit string. Thus, in the scheme, $Y + i = N\|\langle 1 + i\rangle$. Function msb_n, which is needed to compute the final and possibly incomplete ciphertext block C_ℓ, maps a string of length $\ge n$ to its n-bit prefix. The final step in the scheme is then to compute the function H on H and $C = C_1\| \ldots \|C_\ell$ and xor it to the output of the block cipher on input Y. To simplify the technical descriptions in our proofs, we define the ciphertext as consisting of the tag prepended to the output of the counter-mode encryption.

GCM, as proposed by McGrew and Viega [24] and standardized by NIST [12], is obtained by instantiating the block cipher with AES, so that $\lambda = \kappa = 128$. The nonce length (in the standardized version) is $\nu = 96$. The hash function H is based on polynomial evaluation. The specifics do not matter for us. For our security analysis, all we need is that H is an ϵ-almost XOR-universal hash function (according to our definition of Section 2) for some $\epsilon\colon \mathbb{N}\times\mathbb{N} \to [0,1]$. McGrew and Viega [24, Lemma 2] show that H has this property for $\epsilon(m,n) = (\lceil m/\lambda \rceil + \lceil n/\lambda \rceil + 1)/2^\lambda$.

CAU has fixed-length nonces, reflecting the standardized version of GCM in which $\nu = 96$. While the original scheme allows variable-length nonces [24], IOM [18] showed that the original security proof was flawed for variable-length nonces and the claimed security bounds did not hold.

$\underline{\mathsf{XCAU.Enc}^{\mathrm{E},\mathrm{E}^{-1}}(K\|L,N,M,H)}$	$\underline{\mathsf{XCAU.Dec}^{\mathrm{E},\mathrm{E}^{-1}}(K\|L,N,T\|C,H)}$
$\ell \leftarrow \lceil\, \|M\|/\lambda \,\rceil$	$\ell \leftarrow \lceil\, \|M\|/\lambda \,\rceil - 1$
$M_1\| \ldots \|M_\ell \leftarrow M$ // block length λ	$C_1\| \ldots \|C_\ell \leftarrow C$ // block length λ
$r \leftarrow \|M_\ell\|$ // last block length	$r \leftarrow \|C_\ell\|$ // last block length
$G \leftarrow L \oplus \mathrm{E}(K,L)$; $Y \leftarrow N\|\langle 1\rangle$	$G \leftarrow L \oplus \mathrm{E}(K,L)$; $Y \leftarrow N\|\langle 1\rangle$
For $i = 1$ to $\ell - 1$	$T' \leftarrow \mathsf{H}(G,H,C) \oplus L \oplus \mathrm{E}(K, L \oplus (Y+0))$
$\quad C_i = M_i \oplus L \oplus \mathrm{E}(K, L \oplus (Y+i))$	If $T \neq T'$ then return $\perp$
$C_\ell \leftarrow M_\ell \oplus \mathrm{msb}_r(L \oplus \mathrm{E}(K, L \oplus (Y+\ell)))$	For $i = 1$ to $\ell - 1$
$C \leftarrow C_1\| \ldots \|C_\ell$	$\quad M_i \leftarrow C_i \oplus L \oplus \mathrm{E}(K, L \oplus (Y+i))$
$T \leftarrow \mathsf{H}(G,H,C) \oplus L \oplus \mathrm{E}(K, L \oplus (Y+0))$	$M_\ell \leftarrow C_\ell \oplus \mathrm{msb}_r(L \oplus \mathrm{E}(K, L \oplus (Y+\ell)))$
Return $T\|C$	Return $M_1\| \ldots \|M_\ell$

Figure 5: Encryption scheme $\mathsf{XCAU} = \mathbf{XCAU}[\mathsf{H},\kappa,\lambda,\nu]$. **Left:** Encryption algorithm XCAU.Enc. **Right:** Decryption algorithm XCAU.Dec.

RCAU. The TLS Working Group introduced a specific usage mode of GCM in recent draft versions of TLS 1.3 to prevent the scheme from evaluating the block cipher on the same inputs in each session. This countermeasure is described as computing an additional ν bits of key material in the key-derivation phase, and using these to mask the ν-bit nonce given to GCM.

In order to analyze the effectiveness of this countermeasure, we take a different perspective, casting the method as specifying a new symmetric encryption scheme in which the mask becomes part of the key. Formally, as before, let $\kappa, \lambda, \nu \geq 1$ be integers representing the cipher key length, block length and nonce length, where $\nu \leq \lambda - 2$. Let $\mathsf{H}\colon \{0,1\}^\lambda \times (\{0,1\}^* \times \{0,1\}^*) \to \{0,1\}^\lambda$ be an ϵ-XOR universal hash function. We associate to these the symmetric encryption scheme $\mathsf{RCAU} = \mathbf{RCAU}[\mathsf{H},\kappa,\lambda,\nu]$ whose encryption and decryption algorithms are specified in Fig. 4. The scheme has key length $\mathsf{RCAU.kl} = \kappa + \nu$, cipher key length $\mathsf{RCAU.ckl} = \kappa$ and block length $\mathsf{RCAU.bl} = \lambda$. It has nonce space $\mathsf{RCAU.NS} = \{0,1\}^\nu$ and ciphertext length function $\mathsf{RCAU.cl}(\cdot)$ defined by $\mathsf{RCAU.cl}(m) = m + \lambda$. Note that the key length is $\kappa + \nu$, while that of CAU was κ. The definition of $Y + i$ is as per Equation (1), so $Y + i = (N \oplus L)\|\langle 1 + i\rangle$.

XCAU. We suggest a different scheme to achieve the multi-user security goal targeted by RCAU. Recall that if $\mathrm{E}\colon \{0,1\}^\kappa \times \{0,1\}^\lambda \to \{0,1\}^\lambda$ is a block cipher, then $\mathrm{EX}\colon \{0,1\}^{\kappa+\lambda} \times \{0,1\}^\lambda \to \{0,1\}^\lambda$ is the block cipher defined by $\mathrm{EX}(K\|L, X) = L \oplus \mathrm{E}(K, L \oplus X)$. This can be viewed as strengthening E using an Even-Mansour technique [13]. This was suggested by Rivest as a key-extension method for DES and first analyzed by Kilian and Rogaway [19]. We then simply use EX in place of E in the basic CAU. Formally, as before, let $\kappa, \lambda, \nu \geq 1$ be integers representing the cipher key length, block length and nonce length, where $\nu \leq \lambda - 2$. Let $\mathsf{H}\colon \{0,1\}^\lambda \times (\{0,1\}^* \times \{0,1\}^*) \to \{0,1\}^\lambda$ be an ϵ-XOR universal hash function. We associate to these the symmetric encryption scheme $\mathsf{XCAU} = \mathbf{XCAU}[\mathsf{H},\kappa,\lambda,\nu]$ whose encryption and decryption algorithms are specified in Fig. 5. The scheme has key length $\mathsf{XCAU.kl} = \kappa + \lambda$, cipher key length $\mathsf{XCAU.ckl} = \kappa$ and block length $\mathsf{XCAU.bl} = \lambda$. It has nonce space $\mathsf{XCAU.NS} = \{0,1\}^\nu$ and ciphertext length function $\mathsf{XCAU.cl}(\cdot)$ defined by $\mathsf{XCAU.cl}(m) = m + \lambda$. Note that the key length is $\kappa + \lambda$, while that of RCAU was $\kappa + \nu$. The definition of $Y + i$ is as per Equation (1), so $Y + i = N\|\langle 1 + i\rangle$.

Our analysis of this scheme builds on the work of Kilian and Rogaway, but analyzes the construction directly in the multi-user setting. We believe that the bounds can be further improved along the lines of Mouha and Luykx's work [25], but this does not affect the terms we are most interested in.

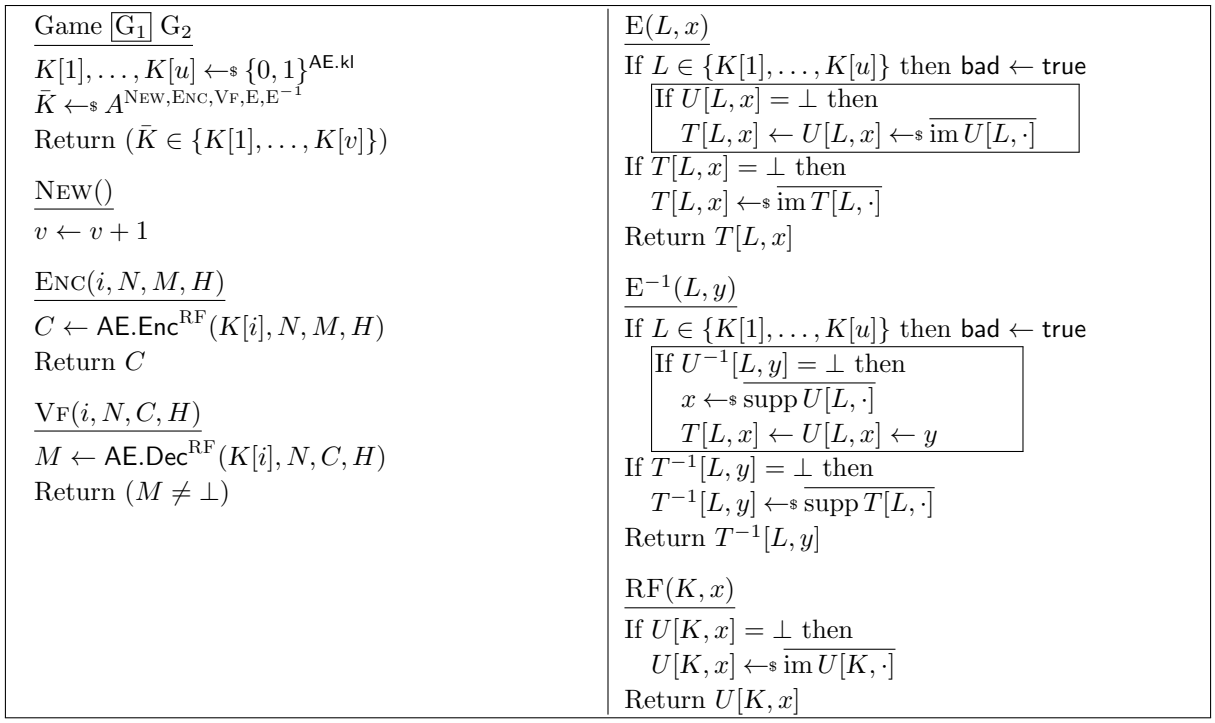

Figure 6: Intermediate games for decoupling oracle RF from oracles E and E^{-1}, in the proof of Theorem 5.1.

---

5 Key-Recovery Security

The multi-user security differences between the schemes are most easily seen in the case of security against key recovery, so we start there.

5.1 Security of CAU

We show that the multi-user kr advantage scales linearly in the number of adversarial evaluations of the ideal cipher (corresponding to offline evaluations of the blockcipher in practice) and the number of user instances. We give both an upper bound (security proof) and lower bound (attack) on the kr-advantage to show this, beginning with the former.

Theorem 5.1 *Let $\kappa, \lambda, \nu \geq 1$ be such that $\nu \leq \lambda - 2$. Let $\mathsf{H}\colon \{0,1\}^{\lambda} \times (\{0,1\}^* \times \{0,1\}^*) \to \{0,1\}^{\lambda}$ be a family of functions. Let $\mathsf{CAU} = \mathbf{CAU}[\mathsf{H}, \kappa, \lambda, \nu]$. Let A be an adversary that makes at most u queries to its* New *oracle and p queries to its* E *and* E^{-1} *oracles. Then*

$$\mathsf{Adv}^{\text{mu-kr}}_{\mathsf{CAU}}(A) \leq \frac{u(p+1)}{2^{\kappa}}.$$

Proof: We use the code-based game-playing technique of BR [6]. Without loss of generality, we assume that the adversary A does not input invalid user identifiers $i \notin \{1, \ldots, v\}$ to Enc or Vf, and does not re-use nonces in encryption queries. We also assume that A does not verify correct ciphertexts obtained from Enc at its Vf oracle. These restrictions allow us to simplify the descriptions of the games, and any arbitrary adversary A can be translated into an adversary A'

that adheres to these restrictions and makes at most the same number of queries as A. Our proof proceeds in a sequence of games.

The first step in the proof is to rewrite game $\mathbf{G}^{\text{mu-kr}}_{\mathsf{CAU}}(A)$ syntactically by introducing an additional oracle RF that implements the forward evaluation of the ideal cipher for the algorithms CAU.Enc and CAU.Dec. This is sufficient as encryption and decryption in CAU never query E^{-1}. We call this game G_0, but do not explicitly describe it here as it is obtained easily from $\mathbf{G}^{\text{mu-kr}}_{\mathsf{CAU}}(A)$.

We then rewrite the game in the form of G_1, which is described in Fig. 6. Oracle RF now samples the ideal cipher for the keys used in the encryption using the map $U[\cdot,\cdot]$, and oracles E and E^{-1} are adapted such that, for keys that are used by honest instances, they sample the map $T[\cdot,\cdot]$ consistently with $U[\cdot,\cdot]$. We also change how keys for new user instances are sampled. Instead of sampling them within NEW, we sample all keys $K[1],\dots K[u]$ in the beginning of the game. All these changes do not change the behavior of the game and the adversary advantage. We then introduce a flag bad that is set when adversary A queries one of the oracles E or E^{-1} with a key that is also used by an honest user instance.

The next game G_2 modifies the way in which the responses for the E, E^{-1}, and RF oracles are determined. In particular, we break the consistency between E and E^{-1} on the one hand, and RF on the other hand, by sampling the oracle responses independently. Since all changes appear only after bad has been set, we can relate the games using the Fundamental Lemma from Bellare and Rogaway [6] and proceed by bounding the probability of setting bad.

The probability of bad being set is in fact bounded by $up2^{-\kappa}$. We first argue that the best adversary strategy for provoking bad to be set is non-adaptive: consider an adversary $B = B(A)$ that plays game G_2 and simulates the same game to A. After A terminates, B replays the same queries to G_2. Observe that G_2 evaluates ENC and VF on one side, and E and E^{-1} on the other side, using independent uniform permutations, which is exactly what B emulates to A, and independent of the actual values $K[1],\dots,K[u]$ in G_2. Therefore, the probability of B to provoke bad in G_2 is exactly the same as the one for A, and we can restrict our attention to non-adaptive adversaries.

Consider now a non-adaptive adversary B. This means, in particular, that we can consider the keys $K[1],\dots,K[u]$ as being chosen after B terminates. (During the game, those keys are only used to determine whether bad is set.) During the game, at most p keys are used in calls to E and E^{-1}; denote the set of all keys used in those queries as Q. The condition to set bad can then be written as $\{K[1],\dots,K[u]\}\cap Q \neq \emptyset$. As the u keys are chosen independently and uniformly at random, and the probability for each key to be in Q is bounded by $p/2^\kappa$, the probability for $\{K[1],\dots,K[u]\}\cap Q = \emptyset$ is at least $(1-p/2^\kappa)^u$, which can be lower-bounded by $e^{-\frac{up}{2^\kappa}}$. As $1-x \le e^{-x}$, we obtain that $1-\frac{up}{2^\kappa} \le e^{-\frac{up}{2^\kappa}} \le (1-p/2^\kappa)^u$ and therefore that the probability of $\{K[1],\dots,K[u]\}\cap Q \neq \emptyset$ is upper-bounded by $\frac{up}{2^\kappa}$.

The keys in G_2 only serve as labels, and the game apart from setting the flag bad is independent of their actual values. The only remaining step is to compute the probability of guessing any one of the u keys that are chosen uniformly at random in G_2, which is bounded by $u2^{-\kappa}$. In more detail:

$$\begin{aligned}\mathsf{Adv}^{\text{mu-kr}}_{\mathsf{CAU}}(A) &= \Pr\left[\mathbf{G}^{\text{mu-kr}}_{\mathsf{CAU}}(A)\right] = \Pr[\mathrm{G}_0] = \Pr[\mathrm{G}_1] \\ &\le \Pr[\mathrm{G}_2] + \Pr[\mathsf{bad}|\mathrm{G}_2] \le \frac{u}{2^\kappa} + \frac{up}{2^\kappa} = \frac{u(p+1)}{2^\kappa},\end{aligned}$$

which concludes the proof. ∎

Next we show that the security bound proven in Theorem 5.1 is (almost) tight. We describe an

Adversary A_{u,p_e}

For $i = 1$ to u do

  NEW ; $C_i \leftarrow \text{ENC}(i, 0^\nu, 0^{2\lambda}, \varepsilon)$

For $j = 1$ to $p_e/2$ do

  $y \leftarrow \text{E}([j]_\lambda, 0^\nu \| \langle 2 \rangle)$ // $[j]_\lambda$ is the encoding of integer j as a λ-bit string

  $y' \leftarrow \text{E}([j]_\lambda, 0^\nu \| \langle 3 \rangle)$

  If $(\exists i : C_i[(\lambda+1)..2\lambda] = y$ and $C_i[(2\lambda+1)..3\lambda] = y')$ then return $[j]_\lambda$

Figure 7: Adversary A_{u,p_e} used in Theorem 5.2.

attack (adversary) that achieves the described bound up to a (for realistic parameters small) factor. The adversary is shown in Fig. 7. It is parameterized by a number u of users and an (even) number p_e of queries to E. It first encrypts a short message $0^{2\lambda}$ for each of the u users. Next, it queries the E oracle on the value $0^\nu \| \langle 2 \rangle$, the first block that is used for masking actual plaintext, for up to p_e different keys. As soon as it finds a matching key L for the first block, it simply evaluates $\text{E}(L, 0^\nu \| \langle 3 \rangle)$ and checks for consistency with the second block. If the check succeeds, the adversary outputs the key L, otherwise it tries further keys.

The described attack strategy extends to any application of CAU in which the nonces used in the scheme are the same in each session. As TLS 1.3 uses the sequence number to compute the nonces, a version without the nonce-randomization technique would be susceptible to this attack.

Theorem 5.2 *Let $\kappa, \lambda, \nu \geq 1$ be such that $\nu \leq \lambda - 2$. Let $\mathsf{H}: \{0,1\}^\lambda \times (\{0,1\}^* \times \{0,1\}^*) \to \{0,1\}^\lambda$ be a family of functions. Let $\mathsf{CAU} = \mathbf{CAU}[\mathsf{H}, \kappa, \lambda, \nu]$. Let $u \geq 1$ be an integer and $p_e \geq 2$ an even integer. Associate to them the adversary A_{u,p_e} described in Fig. 7, which makes u queries to* NEW*, $q_e = u$ queries to* ENC *of length 2λ bits, no queries to* VF*, p_e queries to* E*, and no queries to* E^{-1}*. Then*

$$\mathsf{Adv}^{\text{mu-kr}}_{\mathsf{CAU}}(A_{u,p_e}) \geq \mu \cdot \left(1 - e^{-\frac{p_e u}{2^{\kappa+1}}}\right)$$

where

$$\mu = \left(1 - \frac{u(u-1)}{2^{\kappa+1}}\right) \cdot \left(1 - \frac{u(2^\kappa - u)}{2^\lambda(2^\lambda - 1)}\right) .$$

This means that the advantage of A_{u,p_e} scales (almost) linearly with the number of users, and in fact, for values u, p_e such that $up_e/2^{\kappa+1} \leq 1$, the advantage is lower bounded by $\mu \cdot (1 - 1/e) \cdot \frac{p_e u}{2^{\kappa+1}}$. The proof we give can be improved in terms of tightness, for instance, we allow the attack to completely fail if only a single collision occurs between honest users' keys. In particular the factor $(1 - u(u-1)/2^{\kappa+1})$ could be improved especially for large u.

Proof of Theorem 5.2: The probability for any of the $u = q_e$ keys to collide is at most $u(u-1)/2^{\kappa+1}$. In the subsequent steps we compute the probabilities based on the assumption that no user keys generated within NEW collide, which is correct with probability at least $1 - u(u-1)/2^{\kappa+1}$. In more detail, given that we have no collisions of user keys, the adversary uses at least $p_e/2$ attempts to guess any one of $u = q_e$ (uniformly random, without collision) keys from a set of size 2^κ. The probability for each honest user's key to be among the adversary's guesses is $p_e/2^{\kappa+1}$, and so the overall probability for any one of the adversary's attempts to succeed is

$$1 - \left(1 - \frac{p_e}{2^{\kappa+1}}\right)^u \geq 1 - e^{-\frac{p_e u}{2^{\kappa+1}}}.$$

We still need to bound the probability of false positives, that is, keys that were not sampled in a NEW oracle but coincide with the block cipher outputs, and therefore lead to a wrong guess: The probability that the ideal cipher for a specific "wrong" key (out of $2^\kappa - u$) coincides with the ideal cipher for each of the u "correct" keys on both inputs $0^\nu \| \langle 2 \rangle$ and $0^\nu \| \langle 3 \rangle$ is $2^{-\lambda}(2^\lambda - 1)^{-1}$. The existence of such a colliding key can be bounded using the Union Bound to be at most $u(2^\kappa - u)/(2^\lambda(2^\lambda - 1))$, so the probability that no such collision exists is at least $1 - u(2^\kappa - u)/(2^\lambda(2^\lambda - 1))$. Overall, we obtain the stated bound. ▌

Evaluating the formula for realistic values for GCM in TLS 1.3, we set $\kappa = 128$. We allow the adversary to make $p_e = 2^{64}$ evaluations of the block cipher. We estimate the number of TLS sessions per day as 2^{40}, which leaves us a security margin of roughly 2^{24}. While this means that on expectation the attack still needs 2^{24} days to recover a single key, it is important to recall that this estimate is obtained under the strong assumption that AES behaves like an ideal cipher.

5.2 Security of RCAU

RCAU aims to avoid the attack strategy described in Section 5.1 by randomizing the nonce before it is used in the block cipher. Here we assess whether the measure succeeds, again first upper-bounding adversary advantage via a proof, then lower-bounding it via an attack.

In contrast to the bound for CAU, the bound for RCAU depends on more parameters, and we only prove the result for adversaries that do not make verification queries. Proof and bounds are more complicated since decoupling the block-cipher oracles E and E^{-1} from the oracles ENC and VF is more intricate.

The proof in the proceedings version of this paper [7] is incomplete. Moreover, even the extended proof given here applies only to adversaries that do not make verification queries.

Theorem 5.3 *Let $\kappa, \lambda, \nu \geq 1$ be such that $\nu \leq \lambda - 2$. Let $\mathsf{H}\colon \{0,1\}^\lambda \times (\{0,1\}^* \times \{0,1\}^*) \to \{0,1\}^\lambda$ be a family of functions. Let $\mathsf{RCAU} = \mathbf{RCAU}[\mathsf{H}, \kappa, \lambda, \nu]$. Let A be an adversary that makes at most u queries to its* NEW *oracle, q_e queries per user instance to its* ENC *oracle with messages of length at most ℓ_{bit} bits, no queries to its* VF *oracle, p_e queries to its* E *oracle, and p_i queries to its* E^{-1} *oracle. Then*

$$\mathsf{Adv}^{\text{mu-kr}}_{\mathsf{RCAU}}(A) \leq \frac{u(u+1)}{2^{\kappa+1}} + \frac{c\gamma \cdot p_i}{2^\kappa - p} + \frac{(c+1)\gamma \cdot up(q_e\ell_{\mathrm{blk}} + 1)}{(2^\kappa - p)(2^\lambda - p)} + \frac{(\gamma+1) \cdot upq_e}{(2^\kappa - p)(2^\nu - pq_e)}$$
$$+ \frac{\gamma \cdot upq_e \cdot \ell_{\mathrm{blk}}}{(2^\kappa - p)(2^\lambda - (q_e\ell_{\mathrm{blk}} + 1))} + (p + uq_e\ell_{\mathrm{blk}}) \left(\frac{1}{2^\lambda - uq_e\ell_{\mathrm{blk}}}\right)^c \binom{uq_e\ell_{\mathrm{blk}} + 1}{c+1}, \quad (2)$$

where $\ell_{\mathrm{blk}} = \lceil \ell_{\mathrm{bit}}/\lambda \rceil + 1$, for any fixed $c \in \mathbb{N}$, and for $\gamma = [(1 - u/(2^\kappa - p))(1 - q_e\ell_{\mathrm{blk}}/2^\lambda)]^{-1}$.

The bound stated in the theorem calls for discussion. We begin by stating a corollary that reformulates the bound under reasonable assumptions on the queries. The most restrictive assumption is $pq_e \leq 2^{\nu-1}$, which for the parameters in TLS 1.3 and $p = 2^{64}$ means that at most 2^{31} packets are encrypted with the same key in a given TLS connection. (With maximum-size TLS fragments this still amounts to $\sim$ 30TB.)

Corollary 5.4 *Let all variables be as defined in Theorem 5.3, and assume that $2u + p, 2p \leq 2^\kappa$, $p, uq_e\ell_{\mathrm{blk}} < 2^{\lambda-1}$, and $pq_e < 2^{\nu-1}$. Then*

$$\mathsf{Adv}^{\text{mu-kr}}_{\mathsf{RCAU}}(A) \leq \frac{u^2 + 8p_i}{2^\kappa} + \frac{p \cdot uq_e\ell_{\mathrm{blk}}}{2^{\kappa+\lambda-6}} + \frac{p \cdot uq_e}{2^{\kappa+\nu-4}} + \frac{(p + uq_e\ell_{\mathrm{blk}})(uq_e\ell_{\mathrm{blk}} + 1)^3}{2^{-2\lambda}}\ .$$

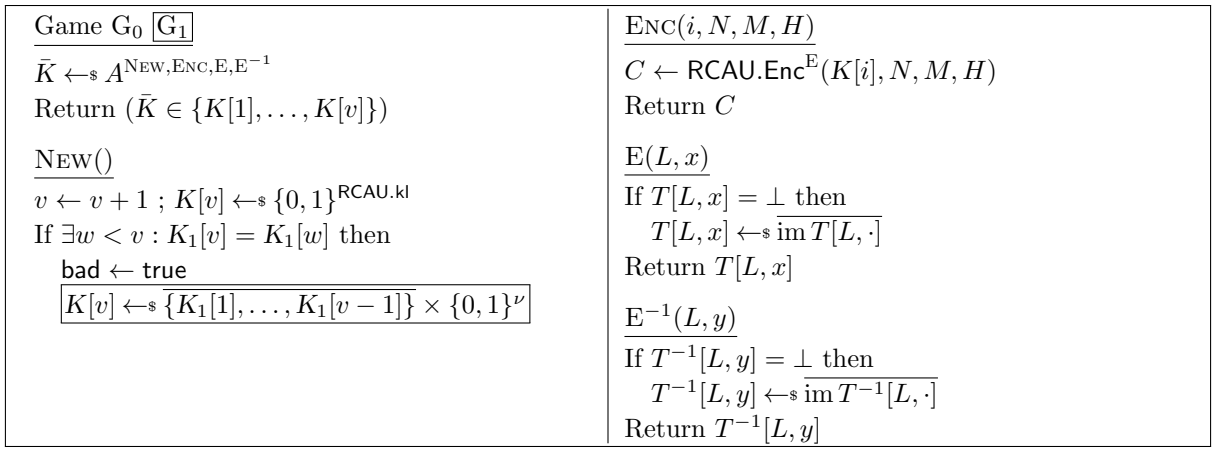

Figure 8: Game G_0 sets bad in case a collision occurs between the block-cipher keys of honest user instances. Game G_1 samples the keys so that no such collision occurs.

For a comparison with CAU, recall that the bound in Theorem 5.1 was $\frac{u(p+1)}{2^\kappa}$. In the bound of Corollary 5.4, however, all terms that involve the product of u and p in the numerator have a denominator that grows exponentially in two parameters, so for realistic parameter values, those terms are significantly smaller than the bound of Theorem 5.1. Concerning the first term, the bound in Corollary 5.4 is better if $p \gg u$. (If p is not large, the bound of Theorem 5.1 is anyway sufficient.)

Proof of Theorem 5.3: The proof proceeds, as in the case of Theorem 5.1, by decoupling the block-cipher oracles E and E^{-1} available to the adversary from the block cipher used in RCAU.Enc and accessible via Enc, but this time we have to be more cautious: we cannot just give up when the adversary uses one of the user keys in a call to the oracles E or E^{-1}; this would ruin our bound.

As in Theorem 5.1, we restrict our attention to adversaries A that do not use invalid user identifiers and do not re-use nonces. This simplifies game G_0, described in detail in Fig. 8 in comparison to $\mathbf{G}^{\text{mu-kr}}_{\mathsf{RCAU}}(A)$ as we can drop the checks. We also add the flag bad and set it to true if, for a new user instance, the first κ bits of the newly sampled key, that is, the part of the key that is used as a the key of the block cipher, coincide with the corresponding part of a key of an existing user instance. We use the following notation: for a user instance $i \in \{1, \ldots, v\}$, the key is split in two parts $K[i] = K_1[i] \| K_2[i]$ with $K_1[i] \in \{0,1\}^\kappa$ and $K_2[i] \in \{0,1\}^\nu$, where $K_1[i]$ is used as a key to the block cipher and $K_2[i]$ is used to mask the input. In the subsequent game G_1, we additionally re-sample the key so that such a collision does not occur. As G_0 and G_1 differ only after bad is set, we can use the Fundamental Lemma of Game Playing to bound the difference in adversary advantage. Since at most u user instances are created, the probability of bad being set can be bounded by $u(u-1)/2^{\kappa+1}$, and therefore $|\Pr[G_0] - \Pr[G_1]| \leq u(u-1)/2^{\kappa+1}$. In the subsequent steps, we can therefore sample the keys as in G_1, namely without collisions in the first part.

The next step is analogous to the beginning of Theorem 5.1 and introduces an additional oracle RF that is accessed by the encryption algorithm instead of E. This new oracle RF can be thought of as an additional copy of E, it internally manages a map $U[\cdot, \cdot]$ analogously to the map $T[\cdot, \cdot]$ in E and E^{-1}, and the oracles are re-defined to keep the maps synchronized. Intuitively, the lazy sampling of the block cipher is now performed using both maps, where $T[\cdot, \cdot]$ is filled in calls to E and E^{-1}, and $U[\cdot, \cdot]$ is filled in RF. The oracles make sure that the maps stay consistent; this is explained in more

Game $\boxed{\mathrm{G}_2}$ G_3

$\bar{K} \leftarrow\!\!\$\ A^{\textsc{New},\textsc{Enc},\mathrm{E},\mathrm{E}^{-1}}$

Return $(\bar{K} \in \{K[1], \ldots, K[v]\})$

$\underline{\textsc{New}()}$

$v \leftarrow v + 1$

$K[v] \leftarrow\!\!\$\ \overline{\{K_1[1], \ldots, K_1[v-1]\}} \times \{0,1\}^{\nu}$

$\underline{\textsc{Enc}(i, N, M, H)}$

$C \leftarrow \mathsf{AE.Enc}^{\mathrm{RF}}(i, N, M, H)$

Return C

$\underline{\mathrm{E}(L, x)}$

If $T[L, x] = \bot$ then

  $T[L, x] \leftarrow\!\!\$\ \overline{\mathrm{im}\, T[L, \cdot]}$

  If $\exists j : K_1[j] = L$ then

    If $x = 0^{\lambda}$ and $U[j, 0^{\lambda}] \neq \bot$ then

      $T[L, 0^{\lambda}] \leftarrow U[j, 0^{\lambda}]$

    If $x \neq 0^{\lambda}$ and $U[j, K_2[j] \oplus x] \neq \bot$ then

      $\mathsf{bad} \leftarrow \mathsf{true}$; $\boxed{T[L, x] \leftarrow U[j, K_2[j] \oplus x]}$

    If $T[L, x] \in \mathrm{im}\, U[j, \cdot]$ then

      $\mathsf{bad} \leftarrow \mathsf{true}$

      $\boxed{T[L, x] \leftarrow\!\!\$\ \overline{\mathrm{im}\, T[L, \cdot] \cup \mathrm{im}\, U[j, \cdot]}}$

Return $T[L, x]$

$\underline{\mathrm{E}^{-1}(L, y)}$

If $T^{-1}[L, y] = \bot$ then

  $x \leftarrow\!\!\$\ \overline{\mathrm{supp}\, T[L, \cdot]}$

  If $\exists j : K_1[j] = L$ then

    If $U[j, 0^{\lambda}] = y$ then $x \leftarrow 0^{\lambda}$

    If $y \in U[j, \overline{\{0^{\lambda}\}}]$ then

      $\mathsf{bad} \leftarrow \mathsf{true}$; $\boxed{x \leftarrow U^{-1}[j, y] \oplus K_2[j]}$

    If $x \in \mathrm{supp}\, U[j, \cdot]$ then

      $\mathsf{bad} \leftarrow \mathsf{true}$

      $\boxed{x \leftarrow\!\!\$\ \overline{\mathrm{supp}\, T[L, \cdot] \cup \mathrm{supp}\, U_{\$}[j, \cdot]}}$

  $T[L, x] \leftarrow y$

Return $T^{-1}[L, y]$

$\underline{\mathrm{RF}(i, x)}$

If $U[i, x] = \bot$ then

  $U[i, x] \leftarrow\!\!\$\ \overline{\mathrm{im}\, U[i, \cdot]}$

  If $x = 0^{\lambda}$ and $T[K_1[i], 0^{\lambda}] \neq \bot$ then

    $U[i, 0^{\lambda}] \leftarrow T[K_1[i], 0^{\lambda}]$

  If $x \neq 0^{\lambda}$ and $T[K_1[i], K_2[i] \oplus x] \neq \bot$ then

    $\mathsf{bad} \leftarrow \mathsf{true}$; $\boxed{U[i, x] \leftarrow T[K_1[i], K_2[i] \oplus x]}$

  If $U[i, x] \in \mathrm{im}\, T[K_1[i], \cdot]$ then

    $\mathsf{bad} \leftarrow \mathsf{true}$

    $\boxed{U[i, x] \leftarrow\!\!\$\ \overline{\mathrm{im}\, T[K_1[i], \cdot] \cup \mathrm{im}\, U[i, \cdot]}}$

Return $U[i, x]$

Figure 9: Game G_2 is almost the same as G_1 but changes the sampling procedure in E, E^{-1}, and RF, as well as the addressing of the keys in RF. Game G_3 differs in that the consistency between T and U is dropped.

---

detail below. In contrast to $T[\cdot, \cdot]$, however, $U[\cdot, \cdot]$ only contains values corresponding to honest user instances and is therefore addressed by user indices instead of keys, i.e., $U[i, \cdot]$ as opposed to $T[K_1[i], \cdot]$. We apply this change also to the encryption algorithm and the oracle Enc, where (by slight abuse of notation) we generally use the user index i instead of the key $K[i]$. Another modification changes where the block-cipher inputs are randomized; in game G_1 this occurs within the algorithm $\mathsf{RCAU.Enc}$, in game G_2 we take the perspective that this becomes part of the oracle RF—and even the map $U[\cdot, \cdot]$. Consequently, we replace the encryption algorithm $\mathsf{RCAU.Enc}^{\mathrm{E}}$ in Enc by the non-randomized variant $\mathsf{AE.Enc}^{\mathrm{RF}}$. This is only a syntactic change. To summarize, in game G_1, the algorithm $\mathsf{RCAU.Enc}$ is called in oracle Enc with the key $K[i]$ for user i as parameter, algorithm $\mathsf{RCAU.Enc}$ then uses the first κ bits of $K[i]$ as key in calls to E and the last ν bits for randomizing the first ν bits of inputs to E (except for 0^{λ}, which is not randomized). In game G_2, the algorithm $\mathsf{AE.Enc}$ is called in oracle Enc with the user index i as parameter.[1] Algorithm $\mathsf{AE.Enc}$ also uses the index i in calls to oracle RF, and oracle RF uses the appropriate key $K_1[i]$ as first argument to $U[\cdot, \cdot]$ and masks the first ν bits of input x with $K_2[i]$, as previously done within $\mathsf{RCAU.Enc}$. Here, and in the following, we write $K_2[i] \oplus x$ to mean that $K_2[i] \in \{0,1\}^{\nu}$ is xored to the first ν bits of $x \in \{0,1\}^{\lambda}$. The oracle RF also handles the input 0^{λ} appropriately, namely without randomization.

[1] This is by slight abuse of notation for the algorithms.

More technically, the outputs of the oracles E, E^{-1}, and RF are first sampled as if the maps $T[\cdot,\cdot]$ and $U[\cdot,\cdot]$ were independent; namely uniformly from $\overline{\mathrm{im}\, T[L,\cdot]}$ in E, from $\overline{\mathrm{supp}\, T[L,\cdot]}$ in E^{-1}, and from $\overline{\mathrm{im}\, U[i,\cdot]}$ in RF. Then, the oracles check whether the sampled value violates the equation $T[K_1[i], K_2[i] \oplus x] = U[i,x]$ for all $x \neq 0^\lambda$ that must be preserved. In that case, the response is set to the consistent value (if it is determined by the previous state) or re-sampled according to the condition (if it is not determined but the newly sampled value violates the condition). In all cases where a violation occurs, the flag bad is set. The value 0^λ is treated specially because it is the only value not randomized in RF, and therefore $T[K_1[i], 0^\lambda] = U[i, 0^\lambda]$ must be satisfied; the oracles assure that this condition is preserved but unlike for $x \neq 0^\lambda$ they do not set the flag bad if the value is copied from the respective other map. Overall, all changes from G_1 to G_2, which is explicitly described in Fig. 9, only change the game syntactically and therefore $\Pr[\mathrm{G}_1] = \Pr[\mathrm{G}_2]$.

In game G_3, which is also described in Fig. 9 and does not contain the boxed code, so the sampled outputs in the oracles E, E^{-1}, and RF, are not overwritten after bad is set. This is the only modification, and by the Fundamental Lemma of Game Playing, we obtain that $|\Pr[\mathrm{G}_2] - \Pr[\mathrm{G}_3]| \leq \Pr[\mathsf{bad} \text{ in } \mathrm{G}_3]$.

The next game hop requires an additional definition. In particular, we define the set

$$
\begin{aligned}
\mathrm{GK}^v_{T,U} = \Big\{ & (\tilde{K}[1], \ldots, \tilde{K}[v], \tilde{U}[1,0^\lambda], \ldots, \tilde{U}[v,0^\lambda]) \in \left(\{0,1\}^{\mathsf{RCAU.kl}}\right)^v \times \left(\{0,1\}^\lambda\right)^v : \\
& \tilde{K}_1[i] \neq \tilde{K}_1[j] \text{ for all } i,j \in \{1,\ldots,v\}, i \neq j \;, && (3) \\
& \tilde{U}[i,x] = U[i,x] \text{ for all } i \in \{1,\ldots,v\}, x \in \mathrm{supp}\, U[i,\cdot] \setminus \{0^\lambda\} \;, \\
& \text{and } \tilde{U}[i,0^\lambda] \notin \tilde{U}[i, \overline{\{0^\lambda\}}] \text{ for all } i \in \{1,\ldots,v\} \;, && (4) \\
& \text{and } \tilde{U}[i,0^\lambda] = T[\tilde{K}_1[i], 0^\lambda] \text{ for all } i \in \{1,\ldots,v\} \text{ with } T[\tilde{K}_1[i], 0^\lambda] \neq \perp \;, && (5) \\
& \text{and } \mathrm{im}\, T[\tilde{K}_1[i],\cdot] \cap \mathrm{im}\, \tilde{U}[i,\cdot] = \emptyset \text{ for all } i \in \{1,\ldots,v\} \;, && (6) \\
& \text{and } \{(i,x) : T[\tilde{K}_1[i], \tilde{K}_2[i] \oplus x] \neq \perp\} \cap \mathrm{supp}\, \tilde{U}[\cdot,\cdot] = \emptyset \Big\} && (7)
\end{aligned}
$$

of so-called *good keys*, which are those values for the user keys that do not provoke an inconsistency between the maps T and U. (The values $\tilde{U}[1,0^\lambda], \ldots, \tilde{U}[u,0^\lambda]$ can be seen as keys of the AXU hash function.)

The set $\mathrm{GK}^v_{T,U}$ deserves additional discussion. On a high level, $\mathrm{GK}^v_{T,U}$ contains exactly those tuples $(K[1], \ldots, K[v], U[1,0^\lambda], \ldots, U[v,0^\lambda])$ for which bad would not be set in G_3. In particular, due to line (3) there is no collision among the *first parts* of the keys, namely those parts that are used as a key to the block cipher in RCAU, like they are sampled in the New oracle. Line (4) simply ensures that $\tilde{U}$ defines a proper permutation. Second, line (5) enforces the consistency of $U[\cdot,0^\lambda]$ and $T[K[\cdot],0^\lambda]$, if that latter value is defined, as ensured in G_2. Third, due to line (6) a key $K_1[j]$ can only have the value $K_1[j] = L$ if there are no collisions between $T[K_1[j],\cdot]$ and $U[j,\cdot]$, a condition that occurs in E, E^{-1}, and RF. Line (7) prevents all key values that provoke an *input collision* between $T[K_1[j], K_2[j] \oplus \cdot]$ and $U[j,\cdot]$. Overall, the conditions in the set $\mathrm{GK}^v_{T,U}$ correspond exactly to the conditions of bad being set in G_2 and G_3.

In game G_4, described in Fig. 10, the description of the oracles E, E^{-1}, and RF is simplified. There is also a further change to these oracles. After the return value of the respective oracle has been sampled, the game samples keys $(\tilde{K}[1], \ldots, \tilde{K}[v])$ and block-cipher values $(\tilde{U}[1,0^\lambda], \ldots, \tilde{U}[v,0^\lambda])$ from the set $\mathrm{GK}^v_{T,U}$. The flag bad is then also set when, for some value $y \in \{0,1\}^\lambda$, more than c pairs of index and block $(i,x) \in \{1,\ldots,v\} \times \{0,1\}^\lambda$ satisfy $U[i,x] = y$ or $\tilde{U}[j,0^\lambda] = y$. The exact

Game G_4 $\boxed{\mathrm{G}_5}$

$\bar{K} \leftarrow\!\!\$\ A^{\textsc{New},\textsc{Enc},\mathrm{E},\mathrm{E}^{-1}}$

Return $(\bar{K} \in \{K[1], \ldots, K[v]\})$

$\underline{\textsc{New}()}$

$v \leftarrow v + 1$

$K[v] \leftarrow\!\!\$\ \overline{\{K_1[1], \ldots, K_1[v-1]\}} \times \{0,1\}^\nu$

$\underline{\textsc{Enc}(i, N, M, H)}$

$C \leftarrow \mathsf{AE.Enc}^{\mathrm{RF}}(i, N, M, H)$

Return C

$\underline{\mathrm{E}(L, x)}$

If $T[L, x] = \bot$ then

  $T[L, x] \leftarrow\!\!\$\ \overline{\mathrm{im}\, T[L, \cdot]}$

  If $\exists j : K_1[j] = L$ then

    If $x = 0^\lambda$ and $U[j, 0^\lambda] \neq \bot$ then

      $T[L, 0^\lambda] \leftarrow U[j, 0^\lambda]$

    If $(x \neq 0^\lambda$ and $U[j, K_2[j] \oplus x] \neq \bot)$

      or $T[L, x] \in \mathrm{im}\, U[j, \cdot]$ then $\mathsf{bad} \leftarrow \mathsf{true}$

    $(\tilde{K}, \tilde{U}[\cdot, 0^\lambda]) \leftarrow\!\!\$\ \mathrm{GK}^v_{T,U}$

    $\tilde{U}[\cdot, \overline{\{0^\lambda\}}] \leftarrow U[\cdot, \overline{\{0^\lambda\}}]$; $\boxed{U[\cdot, 0^\lambda] \leftarrow \tilde{U}[\cdot, 0^\lambda]}$

    If $\exists y \in \{0,1\}^\lambda : |\tilde{U}^{-1}[\cdot, y]| > c$ then

      $\mathsf{bad} \leftarrow \mathsf{true}$

    $\boxed{K \leftarrow \tilde{K}}$

Return $T[L, x]$

$\underline{\mathrm{E}^{-1}(L, y)}$

If $T^{-1}[L, y] = \bot$ then

  $x \leftarrow\!\!\$\ \overline{\mathrm{supp}\, T[L, \cdot]}$

  If $\exists j : K_1[j] = L$ then

    If $U[j, 0^\lambda] = y$ then $x \leftarrow 0^\lambda$

    If $y \in U[j, \overline{\{0^\lambda\}}]$ or $x \in \mathrm{supp}\, U[j, \cdot]$ then

      $\mathsf{bad} \leftarrow \mathsf{true}$

    $(\tilde{K}, \tilde{U}[\cdot, 0^\lambda]) \leftarrow\!\!\$\ \mathrm{GK}^v_{T,U}$

    $\tilde{U}[\cdot, \overline{\{0^\lambda\}}] \leftarrow U[\cdot, \overline{\{0^\lambda\}}]$; $\boxed{U[\cdot, 0^\lambda] \leftarrow \tilde{U}[\cdot, 0^\lambda]}$

    If $\exists y \in \{0,1\}^\lambda : |\tilde{U}^{-1}[\cdot, y]| > c$ then

      $\mathsf{bad} \leftarrow \mathsf{true}$

    $\boxed{K \leftarrow \tilde{K}}$

  $T[L, x] \leftarrow y$

Return $T^{-1}[L, y]$

$\underline{\mathrm{RF}(i, x)}$

If $U[i, x] = \bot$ then

  $U[i, x] \leftarrow\!\!\$\ \overline{\mathrm{im}\, U[i, \cdot]}$

  If $x = 0^\lambda$ and $T[K_1[i], 0^\lambda] \neq \bot$ then

    $U[i, 0^\lambda] \leftarrow T[K_1[i], 0^\lambda]$

  If $x \neq 0^\lambda$ and $T[K_1[i], K_2[i] \oplus x] \neq \bot$ then

    $\mathsf{bad} \leftarrow \mathsf{true}$

  If $U[i, x] \in \mathrm{im}\, T[K_1[i], \cdot]$ then $\mathsf{bad} \leftarrow \mathsf{true}$

  $(\tilde{K}, \tilde{U}[\cdot, 0^\lambda]) \leftarrow\!\!\$\ \mathrm{GK}^v_{T,U}$

  $\tilde{U}[\cdot, \overline{\{0^\lambda\}}] \leftarrow U[\cdot, \overline{\{0^\lambda\}}]$; $\boxed{U[\cdot, 0^\lambda] \leftarrow \tilde{U}[\cdot, 0^\lambda]}$

  If $\exists y \in \{0,1\}^\lambda : |\tilde{U}^{-1}[\cdot, y]| > c$ then $\mathsf{bad} \leftarrow \mathsf{true}$

  $\boxed{K \leftarrow \tilde{K}}$

Return $U[i, x]$

Figure 10: Game G_4 re-samples the keys $K[i]$ and $U[i, 0^\lambda]$ and introduces a further condition on collisions of values in $U[\cdot, \cdot]$. Game G_5 makes the newly sampled keys persistent.

---

purpose of this modification will become clear in subsequent proof steps; essentially, introducing this event will simplify bounding the overall probability of bad being set, since every output value in the map $T[\cdot, \cdot]$ that is defined in E or E^{-1} can affect at most $c \in \mathbb{N}$ values in $U[\cdot, \cdot]$.

The distribution used in game G_4 for sampling from $\mathrm{GK}^v_{T,U}$ is described as follows: Let

$$\#U(K_1, i) = \begin{cases} 2^\lambda - |\,\mathrm{im}\, U[i, \cdot]|\,, & \text{if } T[K_1[i], 0^\lambda] = \bot\,, \\ 2^\lambda\,, & \text{if } \bot \neq T[K_1[i], 0^\lambda] \notin \mathrm{im}\, U[i, \cdot]\,, \\ 0\,, & \text{otherwise}\,, \end{cases}$$

and $\#U(K_1) = \sum_{i=1}^v \#U(K_1, i)$. Analogously, let

$$\#K_2(K_1, i) = \left|\left\{K_2 \in \{0,1\}^\nu : \{x \in \{0,1\}^\lambda : T[K_1[i], K_2 \oplus x]\} \cap \mathrm{supp}\, U[i, \cdot] = \emptyset\right\}\right|\,,$$

and $\#K_2(K_1) = \sum_{i=1}^v \#K_2(K_1, i)$. The keys $K_1[1], \ldots, K_1[v]$ are then chosen weighted according to $\#K_2(K_1)$ and $\#U(K_1)$, with the values $K_2[1], \ldots, K_2[v]$ and $U[1, 0^\lambda], \ldots, U[v, 0^\lambda]$ uniformly from all *legal* values. The exact meaning of this distribution will become clear below.

Overall, in terms of adversary advantage and in fact independent of the above-described distribution, game G_4 is still equivalent to G_3, the main difference between the two games being that the bad flag is more likely to be set in G_4. The adversary advantage in the two games, however, is unaffected, meaning that $\Pr[\mathrm{G}_4] = \Pr[\mathrm{G}_3]$ and $\Pr[\mathsf{bad} \text{ in } \mathrm{G}_3] \leq \Pr[\mathsf{bad} \text{ in } \mathrm{G}_4]$.

In G_5, which is also described in Fig. 10, we then change the handling of the keys $K[\cdot]$. At the end of each oracle E, E^{-1}, and RF, the key vector $(K[1], \ldots, K[v])$ is set to the values $(\tilde{K}[1], \ldots, \tilde{K}[v])$ sampled from the set $\mathrm{GK}^v_{T,U}$ of good keys. It is *a priori* not clear that this modification does not change the game, which is what we argue in the following. First, it is important to note that the values of the key vector $(K[1], \ldots, K[v], U[1, 0^\lambda], \ldots, U[v, 0^\lambda])$ for which bad is not set directly before sampling from $\mathrm{GK}^v_{T,U}$ are *exactly* those vectors in the set $\mathrm{GK}^v_{T,U}$, as discussed above. Second, we must argue that the conditional distribution, given the previous outputs of the oracles, is the same.

The distribution from which we sample from the set $\mathrm{GK}^v_{T,U}$ is of course precisely chosen to resemble this conditional distribution. Generally, each combination of values $K_1[i]$, $K_2[i]$, and $U[i, 0^\lambda]$, for all $i \in \{1, \ldots, v\}$, which is consistent with the previous outputs of the oracles, is chosen with the same probability. That follows by a counting argument. There is one special case, though: if for a value $K_1[i]$, the value $T[K_1[i], 0^\lambda]$ is defined, then this may either collide with a value in $\operatorname{im} U[i, \cdot]$, in which case this value $K_1[i]$ cannot be chosen. Or the value $T[K_1[i], 0^\lambda]$ does not collide with $\operatorname{im} U[i, \cdot]$, in which case $U[i, 0^\lambda]$ is determined and does not impose any restriction to $K_1[i]$, which is the same as if all 2^λ keys were possible. Therefore, the distribution of the oracle outputs is not changed by re-sampling the keys according to the described distribution from the set $\mathrm{GK}^v_{T,U}$ of good keys. Therefore, $\Pr[\mathsf{bad} \text{ in } \mathrm{G}_5] = \Pr[\mathsf{bad} \text{ in } \mathrm{G}_4]$.

Another syntactic change then leads us to G_6. Instead of re-sampling the keys $(K[1], \ldots, K[v], U[1, 0^\lambda], \ldots, U[v, 0^\lambda])$ at the end of the oracles E, E^{-1}, or RF, we shift this sampling to the beginning. (It seems easier to think about this as shifting it to the beginning of the *next* invocation of any of the oracles.) It is easy to see that the distribution does not change because the set $\mathrm{GK}^v_{T,U}$ does not change between the end of an oracle call and the beginning of the next one. Also, for a new user instance i the set $\mathrm{GK}^v_{T,U}$ does not impose any restrictions on $K[i]$, and therefore they are chosen appropriately uniformly from the set $\{0,1\}^{\mathsf{RCAU.kl}}$. Therefore, $\Pr[\mathsf{bad} \text{ in } \mathrm{G}_6] = \Pr[\mathsf{bad} \text{ in } \mathrm{G}_5]$.

In game G_6, we can now finally bound the probability of an adversary in provoking bad to be set. Before we analyze the probabilities for each of the oracles in detail, we continue with some basic observations. For a key $L_1 \in \{0,1\}^\kappa$ that has m values defined in $T[L_1, \cdot]$ and a user instance i that has n values defined in $U[i, \cdot]$, the number of excluded masks $L_2 \in \{0,1\}^\nu$ is bounded by mn. Due to the specific format of inputs that Enc makes to RF (namely that the trailing $\lambda - \nu$ bits are different), each of the p calls to E or E^{-1} can collide with at most one of the flight of (at most ℓ_{blk}) queries made to RF during an evaluation of Enc, and so for q_e calls to Enc and $K_1[i] = L_1$, there can be at most pq_e excluded values for $K_2[i] \in \{0,1\}^\nu$.

The bound of p queries to E and E^{-1} means that, for at least $2^\kappa - p$ keys $L_1 \in \{0,1\}^\kappa$, no queries $\mathrm{E}(L_1, \cdot)$ and $\mathrm{E}^{-1}(L_1, \cdot)$ have been made at all, and therefore the map $T[L_1, \cdot]$ is completely undefined for such keys $L_1 \in \{0,1\}^\kappa$. Note that for any such key L_1, and any user instance i with $K_1[i] = L_1$, the number of remaining combinations for $K_2[i] \in \{0,1\}^\nu$ and $K[j] \in \{0,1\}^{\kappa+\nu}$, all $j \neq i$, is the same in $\mathrm{GK}^v_{T,U}$. Therefore, for two keys $L_1, L_1' \in \{0,1\}^\kappa$ for which no queries to E and E^{-1} have been made, the probability for $K_1[i] = L_1$ and $K_1[i] = L_1'$ are the same, and therefore bounded by $(2^\kappa - p)^{-1}$. For any key L_1'' for which (at most p) queries to E or E^{-1} have been made, and user instance i with $K_1[i] = L_1''$, the number of possible masks $K_2[i] \in \{0,1\}^\nu$ is at most as large as that

Game G_6

$\bar{K} \leftarrow\!\!\$\ A^{\textsc{New},\textsc{Enc},\mathrm{E},\mathrm{E}^{-1}}$

Return $(\bar{K} \in \{K[1], \ldots, K[v]\})$

$\underline{\textsc{New}()}$

$v \leftarrow v + 1$

$K[v] \leftarrow\!\!\$\ \overline{\{K_1[1], \ldots, K_1[v-1]\}} \times \{0,1\}^\nu$

$\underline{\textsc{Enc}(i, N, M, H)}$

$C \leftarrow \mathsf{AE.Enc}^{\mathrm{RF}}(i, N, M, H)$

Return C

$\underline{\mathrm{E}(L, x)}$

If $T[L, x] = \bot$ then

  $T[L, x] \leftarrow\!\!\$\ \overline{\mathrm{im}\, T[L, \cdot]}$

  If $\exists j : K_1[j] = L$ then

    $K[\cdot], U[\cdot, 0^\lambda] \leftarrow\!\!\$\ \mathrm{GK}^v_{T,U}$

    If $x = 0^\lambda$ and $U[j, 0^\lambda] \neq \bot$ then

      $T[L, 0^\lambda] \leftarrow U[j, 0^\lambda]$

    If $(x \neq 0^\lambda$ and $U[j, K_2[j] \oplus x] \neq \bot)$

      or $T[L, x] \in \mathrm{im}\, U[j, \cdot]$

      or $\exists y \in \{0,1\}^\lambda : |U^{-1}[\cdot, y]| > c$ then

      $\mathsf{bad} \leftarrow \mathsf{true}$

Return $T[L, x]$

$\underline{\mathrm{E}^{-1}(L, y)}$

If $T^{-1}[L, y] = \bot$ then

  $x \leftarrow\!\!\$\ \overline{\mathrm{supp}\, T[L, \cdot]}$

  If $\exists j : K_1[j] = L$ then

    $K[\cdot], U[\cdot, 0^\lambda] \leftarrow\!\!\$\ \mathrm{GK}^v_{T,U}$

    If $U[j, 0^\lambda] = y$ then $x \leftarrow 0^\lambda$

    If $y \in U[j, \overline{\{0^\lambda\}}]$ or $x \in \mathrm{supp}\, U[j, \cdot]$

      or $\exists y \in \{0,1\}^\lambda : |U^{-1}[\cdot, y]| > c$ then

      $\mathsf{bad} \leftarrow \mathsf{true}$

  $T[L, x] \leftarrow y$

Return $T^{-1}[L, y]$

$\underline{\mathrm{RF}(i, x)}$

If $U[i, x] = \bot$ then

  $K[\cdot], U[\cdot, 0^\lambda] \leftarrow\!\!\$\ \mathrm{GK}^v_{T,U}$

  $U[i, x] \leftarrow\!\!\$\ \overline{\mathrm{im}\, U[i, \cdot]}$

  If $x = 0^\lambda$ and $T[K_1[i], 0^\lambda] \neq \bot$ then

    $U[i, 0^\lambda] \leftarrow T[K_1[i], 0^\lambda]$

  If $(x \neq 0^\lambda$ and $T[K_1[i], K_2[i] \oplus x] \neq \bot)$

    or $U[i, x] \in \mathrm{im}\, T[K_1[i], \cdot]$

    or $\exists y \in \{0,1\}^\lambda : |U^{-1}[\cdot, y]| > c$ then

    $\mathsf{bad} \leftarrow \mathsf{true}$

Return $U[i, x]$

Figure 11: In G_6, the re-sampling is shifted to the beginning of the oracles.

for L_1 (where it is maximal). On the other hand, for all $i \in \{1, \ldots, v\}$, masks $M \in \{0,1\}^\nu$, and outputs $U[i, 0^\lambda]$,

$$\mathrm{Pr}_{\vec{K},\vec{U} \leftarrow\!\!\$\ \mathrm{GK}^v_{T,U}}[K_1[i] = L_1'' \wedge \forall j : K_1[j] \neq L_1] \leq \frac{2^\lambda}{2^\lambda - q_e \ell_{\mathrm{blk}}} \cdot \mathrm{Pr}_{\vec{K},\vec{U} \leftarrow\!\!\$\ \mathrm{GK}^v_{T,U}}[K_1[i] = L_1 \wedge \forall j : K_1[j] \neq L_1''] \,,$$

which can be seen as follows. Condition (3) mandates that $K_1[j] \neq K_1[i]$ for all $j \neq i$, and conditions (4) only mandates that $\tilde{U}[i, 0^\lambda]$ does not collide with any value in $U[i, \overline{\{0^\lambda\}}]$, which is independent of the value of $K[i] \in \{0,1\}^{\kappa+\nu}$. Considering the further cases described in $\mathrm{GK}^v_{T,U}$, the conditions in lines (5, 6, 7) are "local" in the sense that for $j \neq i$, they only depend on $K[j]$ and $U[j, 0^\lambda]$, and not on the value of $K[i]$. Condition (5) may assign a larger probability to $K_1[i] = L_1''$ as $T[L_1'', 0^\lambda]$ may be set (and define $U[i, 0^\lambda]$ consistently) while up to $q_e \ell_{\mathrm{blk}}$ values for $U[i, 0^\lambda]$ may be blocked by other values $U[i, \overline{\{0^\lambda\}}]$ in the case of $K_1[i] = L_1$ (where $T[L_1, 0^\lambda]$ is not set). For $K_1[i] = L_1$, the number of combinations of values for $K_2[i]$ and $U[i, 0^\lambda]$ that are valid by conditions (6, 7) is at least as large as for $K_1[i] = L_1''$, as they are trivial when no values in $T[K_1[i], \cdot]$ are defined.

The probability $\mathrm{Pr}_{\vec{K},\vec{U} \leftarrow\!\!\$\ \mathrm{GK}^v_{T,U}}[\exists j : K_1[j] = L_1 | K_1[i] = L_1'']$ is bounded by $u/(2^\kappa - p)$ and therefore

$$\mathrm{Pr}_{\vec{K},\vec{U} \leftarrow\!\!\$\ \mathrm{GK}^v_{T,U}}[K_1[i] = L_1''] \leq \frac{1}{(1 - u/(2^\kappa - p))(1 - q_e \ell_{\mathrm{blk}}/2^\lambda)} \cdot \mathrm{Pr}_{\vec{K},\vec{U} \leftarrow\!\!\$\ \mathrm{GK}^v_{T,U}}[K_1[i] = L_1] \,,$$

so with $\gamma = (1-u/(2^\kappa-p))(1-q_e\ell_{\text{blk}}/2^\lambda)^{-1}$, the probability for $K_1[i] = L''$ is bounded by $\gamma/(2^\kappa-p)$.

We analyze the probability of bad to be set for each of the oracles E, E^{-1}, and RF individually, but postpone the condition on $|U^{-1}[\cdot, y]|$ to the end, since this is the same for all cases:

- We begin with the oracle E, in which bad may be set because either a new output value sampled for the map T is already defined in U, or because the map U already defines an output for the given input value. We consider an invocation of E, assume that bad is not set initially, and bound the probability of it being set during the invocation.

  We first bound the probability for sampling an output value for T that collides with one in U for the corresponding key, namely the condition $T[L, x] \in \operatorname{im} U[j, \cdot]$ for any $j \in \{1, \ldots, u\}$. As at most p queries to E and E^{-1} can be made for each single key L, the line $T[L, x] \leftarrow\!\!\$\ \overline{\operatorname{im} T[L, \cdot]}$ results in sampling the output uniformly at random from at least $2^\lambda - p$ remaining values. Similarly, since, for each of the u user instances, at most q_e queries to Enc have been made, which each incurred at most ℓ_{blk} calls to RF, the overall number of calls to RF, and therefore values defined in $U[j, \cdot]$ is bounded by $u(q_e\ell_{\text{blk}} + 1)$, with the additional blocks $U[i, 0^\lambda]$ per user instance.

  Combining the latter two facts, the probability of sampling for $T[L, x]$ *any* value that satisfies one of the conditions is bounded by $u(q_e\ell_{\text{blk}} + 1)/(2^\lambda - p)$. The fact that bad is not set then implies that for each $y \in \{0,1\}^\lambda$ at most c equations $U[i, x] = y$ for $i \in \{1, \ldots, u\}$ hold. By the Union Bound, we can then bound the probability that any user instance $i \in \{1, \ldots, v\}$ is assigned the key $K_1[i] = L$ by $c\gamma/(2^\kappa - p)$, by the above argument that the probability for each key to be chosen is at most $\gamma/(2^\kappa - p)$.

  We then continue by bounding the probability that $U[j, K_2[j] \oplus x] \neq \bot$. As above, the probability of each individual key to be chosen for an instance $i \in \{1, \ldots, v\}$ is bounded by $\gamma/(2^\kappa - p)$.

  For $T[L, \cdot]$ and each $U[i, \cdot]$, the number of possible masks in $\{0,1\}^\nu$ that are not valid (as in the definition of $\mathrm{GK}^v_{T,U}$) is bounded by pq_e. This is so because there are at most p entries in $T[L, \cdot]$ and at most q_e entries in $U[i, \cdot]$ that have the same trailing $\lambda - \nu$ bits, and each such pair of entries excludes one input mask for $K_1[i] = L$. Out of the at least $2^\nu - pq_e$ remaining valid masks, we choose one uniformly at random, and there are at most q_e possibilities of inducing a collision between the query x and any field in $U[i, \cdot]$ (since again there are only q_e entries with the appropriate trailing $\lambda - \nu$ bits).

  We apply the above argument for u user instances and use the Union bound, by which we conclude an overall bound of $\gamma u q_e \ / \ (2^\kappa - p)(2^\nu - pq_e)$.

- We continue with the oracle $\mathrm{E}^{-1}(L, y)$, in which bad may be set because either a pre-image newly sampled for the map T is already defined in U, or because the adversary calls E^{-1} on a previous output of RF and guesses the correct key. In the invocation of E^{-1}, we again assume that bad is not set initially, and bound the probability of it being set during the invocation.

  We start by bounding the probability for sampling a pre-image value x for $T[L, y]$ that collides with one in U for the corresponding key, namely the condition $x \in \operatorname{supp} U[j, \cdot]$. As above, we bound the probability of each key to be chosen in a user instance by $\gamma/(2^\kappa - p)$. Also, as at most p queries can be made for each single key L, the line $x \leftarrow\!\!\$\ \overline{\operatorname{supp} T[L, \cdot]}$ results in sampling the output uniformly at random from at least $2^\lambda - p$ remaining values. The overall number of calls to RF for a certain $j \in \{1, \ldots, v\}$, and thereby the number values defined in $U[j, \cdot]$ including 0^λ, is bounded by $q_e\ell_{\text{blk}} + 1$.

Combining the latter two facts, for any $j \in \{1, \ldots, v\}$, the probability of sampling *any* value x with $U[j, K_2[j] \oplus x]$ is bounded by $(q_e \ell_{\text{blk}} + 1)/(2^\lambda - p)$. By the Union Bound, we can then bound the probability that any user instance $i \in \{1, \ldots, v\}$ is assigned the key $K_1[i] = L$ by $\gamma \cdot u/(2^\kappa - p)$, which leads to an overall probability bound of $\gamma \cdot u(q_e \ell_{\text{blk}} + 1)$ / $(2^\kappa - p)(2^\lambda - p)$ for each of the at most p queries.

We then also bound the probability that $y \in U[j, \overline{\{0^\lambda\}}]$. By the assumption that bad is not set, there are at most c such values x that satisfy the conditions. Then again by the fact that any specific user key is chosen with probability at most $\gamma/(2^\kappa - p)$ and by the Union Bound, this amounts to success probability at most $c\gamma/(2^\kappa - p)$ per inversion query.

- Finally, we perform the analogous analysis for the oracle RF, in which bad may be set because either a new output value sampled for the map $U[i, \cdot]$ is already defined in T, or because the map T is already defined for the given input value. As above, in the invocation of RF, we again assume that bad is not set initially, and bound the probability of it being set during the invocation.

  We first bound the probability for sampling an output value for U that collides with one in T for the corresponding key, namely the condition $U[i, x] \in \operatorname{im} T[K_1[i], \cdot]$. As, for each user instance $i \in \{1, \ldots, v\}$, at most q_e queries to Enc have been made, which each incurred at most ℓ_{blk} calls to RF, the overall number of calls to RF, and therefore values defined in $U[i, \cdot]$ is bounded by $q_e \ell_{\text{blk}} + 1$, and the line $U[i, x] \leftarrow\!\!\$\, \overline{\operatorname{im} U[i, \cdot]}$ therefore means sampling from at least $2^\lambda - (q_e \ell_{\text{blk}} + 1)$ remaining values.

  Furthermore, at most p queries can be made to E and E^{-1} overall, the number of values defined in $T[\cdot, \cdot]$ is at most p. The probability of each key $K_1[i]$ to be chosen is again bounded by $\gamma/(2^\kappa - p)$, so there are at most $\gamma \cdot p/(2^\kappa - p)$ values in $T[K_1[i], \cdot]$ in expectation, and we can bound the overall probability of $U[i, x] \in \operatorname{im} T[K_1[i], \cdot]$ by $\gamma \cdot p$ / $(2^\kappa - p)(2^\lambda - (q_e \ell_{\text{blk}} + 1))$. By the Union Bound and the fact that the oracle RF is invoked at most $uq_e \ell_{\text{blk}}$ times, we obtain an overall bound of

$$\frac{\gamma \cdot upq_e \cdot \ell_{\text{blk}}}{(2^\kappa - p)(2^\lambda - (q_e \ell_{\text{blk}} + 1))} \ .$$

  We then continue by bounding the probability that $T[K_1[i], K_2[i] \oplus x] \neq \bot$. The probability of each individual key to be chosen for an instance $i \in \{1, \ldots, v\}$ is again bounded by $\gamma/(2^\kappa - p)$. For $T[L, \cdot]$ and each $U[i, \cdot]$, the number of possible masks in $\{0,1\}^\nu$ that are not valid (as in the definition of $\mathrm{GK}^v_{T,U}$) is bounded by pq_e. This is so because there are at most p entries in $T[L, \cdot]$ and at most q_e entries in $U[i, \cdot]$ that have the same trailing $\lambda - \nu$ bits, and each such pair of entries excludes one input mask for $K_1[i] = L$. Out of the at least $2^\nu - pq_e$ remaining valid masks, one is chosen uniformly at random, and there are at most p possibilities of inducing a collision between the query x and any field in $T[K_1[i], \cdot]$. Even more, for the ℓ_{blk} queries made to RF during a call to Enc *together*, at most p matches can occur (since each entry in $T[K_1[i], \cdot]$ has trailing $\lambda - \nu$ bits appropriate to match at most one of those queries).

  Over the total uq_e queries to Enc, using the Union Bound, we conclude an overall bound of $\gamma \cdot uq_e p$ / $(2^\kappa - p)(2^\nu - pq_e)$.

Finally, we consider the condition that formalizes that more than c values in $\operatorname{im} U[\cdot, \cdot]$ collide, in each of the oracles E, E^{-1}, and RF. The probability of a $(c+1)$-multi-collision in U, as described by the condition $\exists y : |U^{-1}[\cdot, y]| > c$, is bounded by $\left(\frac{1}{2^\lambda - uq_e \ell_{\text{blk}}}\right)^c \binom{uq_e \ell_{\text{blk}} + 1}{c+1}$ since each of the values

Adversary A_{p_i}

NEW ; $C \leftarrow \text{ENC}(1, 0^\nu, 0^{2\lambda}, \varepsilon)$

For $j = 1$ to $p_i/2$ do

  $y \leftarrow \text{E}^{-1}([j]_\lambda, C[(\lambda+1)..2\lambda])$ // $[j]_\lambda$ means encoding as λ-bit string

  If $\exists N \in \{0,1\}^\nu : y = N \| \langle 2 \rangle$ then

    If $\text{E}^{-1}([j]_\lambda, C[(2\lambda+1)..3\lambda]) = N \| \langle 3 \rangle$ then

      Return $[j]_\lambda$

Figure 12: Adversary A_{p_i} used in Theorem 5.5.

is sampled from at least $2^\lambda - uq_e\ell_{\text{blk}}$ values. As we re-sample some values in each of the $p + uq_e\ell_{\text{blk}}$ queries to E, E^{-1}, and RF, we apply the Union Bound and obtain another factor $p + uq_e\ell_{\text{blk}}$.

Collecting all the terms that we showed in the above paragraphs, we can bound the probability of setting bad in G_6, over the respective number of oracle calls, by

$$\Pr[\mathsf{bad} \text{ in } \text{G}_6] \le \frac{(c+1)\gamma \cdot up(q_e\ell_{\text{blk}}+1)}{(2^\kappa - p)(2^\lambda - p)} + \frac{2\gamma \cdot upq_e}{(2^\kappa - p)(2^\nu - pq_e)} + \frac{c\gamma \cdot p_i}{(2^\kappa - p)} + \frac{upq_e}{(2^\kappa - p)(2^\nu - pq_e)} + (p + uq_e\ell_{\text{blk}}) \left(\frac{1}{2^\lambda - uq_e\ell_{\text{blk}}}\right)^c \binom{uq_e\ell_{\text{blk}}+1}{c+1} .$$

The proof of the theorem then concludes by combining the above facts as

$$\Pr[\mathbf{G}^{\text{mu-kr}}_{\mathsf{RCAU}}(A)] \le \Pr[\text{G}_1] + \frac{u(u-1)}{2^{\kappa+1}} = \Pr[\text{G}_2] + \frac{u(u-1)}{2^{\kappa+1}} \le \Pr[\text{G}_3] + \Pr[\mathsf{bad} \text{ in } \text{G}_3] + \frac{u(u-1)}{2^{\kappa+1}} \le \Pr[\text{G}_3] + \Pr[\mathsf{bad} \text{ in } \text{G}_6] + \frac{u(u-1)}{2^{\kappa+1}} ,$$

and using the above bound on $\Pr[\mathsf{bad}|\text{G}_6]$.

Finally, as in G_3 the oracles E and E^{-1} are independent of the oracle RF that is used in RCAU, the probability of guessing a key is bounded by $u/2^\kappa$. (To obtain the theorem statement, note that $u(u-1)/2^{\kappa+1} + u/2^\kappa = u(u+1)/2^{\kappa+1}$.) ▌

For realistic parameters, the bound in Theorem 5.3 means that the "best" attack for passive adversaries is now the inversion of a block observed while eavesdropping. In contrast to the attack analyzed in Section 5.1, this attack does not scale in the mass surveillance scenario, because the adversary has to target one specific ciphertext block.

In more detail, the adversary strategy A analyzed in the below lemma and specified in detail in Fig. 12 proceeds as follows. First obtain an encryption of $0^{2\lambda}$ from an honest user. Then brute-force the key by decrypting the first ciphertext block using E^{-1}, checking whether the output satisfies the structure $N \| \langle 2 \rangle$. In case this structure is observed, verify the key by checking if the next block is consistent with an evaluation of E with the same key and plaintext $N \| \langle 3 \rangle$.

Since the described attack strategy applies independently of how the nonces are chosen (prior to the randomization) as long as the value is predictable, the lower bound also applies to the scheme as used in the latest draft of TLS 1.3.

Theorem 5.5 *Let $\kappa, \lambda, \nu \ge 1$ be such that $\nu \le \lambda - 2$. Let $\mathsf{H}: \{0,1\}^\lambda \times (\{0,1\}^* \times \{0,1\}^*) \to \{0,1\}^\lambda$ be a family of functions. Let $\mathsf{RCAU} = \mathbf{RCAU}[\mathsf{H}, \kappa, \lambda, \nu]$. Let $p_i \ge 2$ an even integer and the*

adversary A_{p_i} as described in Fig. 12, which makes 1 *query to each* NEW *and* ENC *(the latter of length* 2λ *bits), no queries to* VF*,* p_i *queries to* E^{-1}*, and no queries to* E*. Then*

$$\mathsf{Adv}^{\text{mu-kr}}_{\mathsf{RCAU}}(A_{p_i}) \geq \mu \cdot p_i \cdot 2^{-\kappa-1},$$

with

$$\mu = 1 - \frac{(2^\kappa - 1)2^\nu}{2^\lambda(2^\lambda - 1)} \,.$$

Proof: Let $K[1]$ be the key sampled during the invocation of NEW in the game. The probability for the block cipher on a key $K \neq K[1]$ to satisfy the first condition is $2^{\nu-\lambda}$, since in the first invocation of E^{-1} the value is sampled uniformly at random and $\lambda - \nu$ bits have to match. The second invocation of E^{-1} has to lead to the correct outcome $N \| \langle 3 \rangle$, the value is drawn uniformly at random from the remaining $2^\lambda - 1$ values not equal to the outcome of the first query. There are 2^κ keys, so by the Union Bound the probability of any key $K \neq K[1]$ to lead to an admissible pattern on the first two blocks is bounded by $(2^\kappa - 1)2^\nu/(2^\lambda(2^\lambda - 1))$.

In the event that no key $K \neq K[1]$ satisfies the above condition, this advantage of adversary A_{p_i} is simply the probability of guessing a uniformly random key of κ bits in $p_i/2$ attempts, as for each key A_{p_i} spends at most 2 queries. This completes the proof. ∎

The attack analyzed in Theorem 5.5 is considerably harder to mount than the one analyzed in Theorem 5.2, because the queries in the Theorem 5.2 attack can be preprocessed and apply to all observed communication sessions equally, whereas in the Theorem 5.5 attack the queries have to be made for a particular session under attack. Still, in the following Section 5.3, we show that at low computational cost for the honest parties, the Theorem 5.5 attack can be made considerably harder.

5.3 Security of XCAU

The term $c\gamma \cdot p_i/(2^\kappa - p)$ in the bound for RCAU originates in the fact that only the input of the block cipher is masked, and inversion queries by the adversaries are not hindered. In the scheme XCAU, an advantage beyond the randomization of the input to derive the hash function key is that the output of the block cipher is masked, which restricts the power of inversion queries to the block cipher considerably.

Our analysis of XCAU is based on combining the analysis of DESX-like input and output whitening in a multi-user setting, and then prove the security of XCAU along the lines of Theorem 6.2. We first prove a multi-user bound for the DESX-like construction. The security goal is described by the games in Fig. 13.

Theorem 5.6 *Let A be an adversary that makes at most u queries to its* NEW*, q queries to its* RF *oracle per user, p queries to its* E *oracle and* E^{-1} *oracles. Then*

$$|\Pr[\mathrm{R}(A)] - \Pr[\mathrm{S}(A)]| \;\leq\; \frac{u \cdot q \cdot p}{2^{\lambda+\kappa+1}}.$$

Proof: We introduce two intermediate games G_0 and G_1 in Fig. 14. Game G_0 is equivalent to game $\mathrm{R}(A)$; the introduction of the additional map $U[\cdot,\cdot]$ is only syntactic as we make sure that it stays consistent with $T[\cdot,\cdot]$ throughout. We also modify the procedures for sampling new values for the maps $U[\cdot,\cdot]$ and $T[\cdot,\cdot]$ such that first we sample a new value such that it is consistent only with the respective map, then check whether it is consistent with the other map, and re-sample consistently

Game $\mathrm{R}(A)$	Game $\mathrm{S}(A)$
$v \leftarrow 0$; $b \leftarrow\!\!\$\ A^{\textsc{New},\mathrm{E},\mathrm{E}^{-1},\mathrm{RF}}$ Return b	$v \leftarrow 0$; $b \leftarrow\!\!\$\ A^{\textsc{New},\mathrm{E},\mathrm{E}^{-1},\mathrm{RF}}$ Return b
$\textsc{New}()$ $v \leftarrow v + 1$ $(K_1[v], K_2[v]) \leftarrow\!\!\$\ \{0,1\}^\kappa \times \{0,1\}^\lambda$	$\textsc{New}()$ $v \leftarrow v + 1$ $K_1[v] \leftarrow\!\!\$\ \{0,1\}^\kappa$

$\mathrm{RF}(i, x)$
If $T[K_1[i], x \oplus K_2[i]] = \perp$ then
  $T[K_1[i], x \oplus K_2[i]] \leftarrow\!\!\$\ \overline{\mathrm{im}\, T[K_1[i], \cdot]}$
Return $T[K_1[i], x \oplus K_2[i]] \oplus K_2[i]$
	$\mathrm{RF}(i, x)$
	If $i \leq v$ and $U[K_1[i], x] = \perp$ then
	  $U[K_1[i], x] \leftarrow\!\!\$\ \overline{\mathrm{im}\, U[K_1[i], \cdot]}$
	Return $U[K_1[i], x]$

$\mathrm{E}(L, x)$
If $T[L, x] = \perp$ then
  $T[L, x] \leftarrow\!\!\$\ \overline{\mathrm{im}\, T[L, \cdot]}$
Return $T[L, x]$
	$\mathrm{E}(L, x)$
	If $T[L, x] = \perp$ then
	  $T[L, x] \leftarrow\!\!\$\ \overline{\mathrm{im}\, T[L, \cdot]}$
	Return $T[L, x]$

$\mathrm{E}^{-1}(L, y)$
If $T^{-1}[L, y] = \perp$ then
  $x \leftarrow\!\!\$\ \overline{\mathrm{supp}\, T[L, \cdot]}$
  $T[L, x] \leftarrow y$
Return $T^{-1}[L, y]$
	$\mathrm{E}^{-1}(L, y)$
	If $T^{-1}[L, y] = \perp$ then
	  $x \leftarrow\!\!\$\ \overline{\mathrm{supp}\, T[L, \cdot]}$
	  $T[L, x] \leftarrow y$
	Return $T^{-1}[L, y]$

Figure 13: Multi-user security for block-cipher key extension. **Left:** Game giving the adversary access to the actual construction. **Right:** Game giving the adversary access to an independent ideal cipher.

---

if we determine that it is not. In G_1, the map $U[\cdot, \cdot]$ is completely independent of the map $T[\cdot, \cdot]$. Both G_0 and G_1 set the flag bad on occasions where the sampling creates inconsistencies between $U[\cdot, \cdot]$ and $T[\cdot, \cdot]$.

The probability of setting the bad flag in G_2 and G_3 can be bounded as follows. We first observe that besides the bad flag, G_3 is equivalent to S. For both G_2 and G_3, as long as bad is not set, all outputs are uniformly distributed among the values that are valid for the respective oracle and key. Moreover, xoring $K_2[i]$ to all inputs or outputs modifies each concrete permutation; however, the distribution of a uniformly random permutation remains unchanged by this operation. Following the definition of Maurer [23], this means that both games G_2 and G_3 with the respective flags bad are *conditionally equivalent* to the game S. (In other words, *conditioned on* bad not being set, the outputs of the games are distributed exactly as in S.)

Subsequently, we can employ Maurer's result [23, Theorem 1] to bound the distinguishing advantage between G_2 and G_3 by the advantage of the best *non-adaptive* distinguisher. As the adversary makes at most p queries to its E and E^{-1} oracles, and uq queries to its RF oracle, there are $u \cdot q \cdot p$ possible combinations of queries that may provoke the flag bad to be set, and each case appears with probability $2^{-\lambda-\kappa-1}$. We conclude the proof via the Union Bound. ▌

Analogously to the previous results on CAU and RCAU, we now analyze the key-recovery security of XCAU.

Game $\boxed{\mathrm{G}_0}$ $\boxed{\mathrm{G}_1}$

$v \leftarrow 0$; $b \leftarrow\!\!\$\ A^{\textsc{New},\mathrm{E},\mathrm{E}^{-1},\mathrm{RF}}$

Return b

$\underline{\textsc{New}}$

$v \leftarrow v + 1$

$(K_1[v], K_2[v]) \leftarrow\!\!\$\ \{0,1\}^\kappa \times \{0,1\}^\lambda$

$\underline{\mathrm{RF}(i, x)}$

If $U[K_1[i], x \oplus K_2[i]] = \bot$ then

  If $T[K_1[i], x \oplus K_2[i]] = \bot$ then

    $U[K_1[i], x \oplus K_2[i]] \leftarrow\!\!\$\ \overline{\mathrm{im}\, U[K_1[i], \cdot]}$

    If $U[K_1[i], x \oplus K_2[i]] \in \mathrm{im}\, T[K_1[i], \cdot]$ then

      $\mathsf{bad} \leftarrow \mathsf{true}$; $\boxed{U[K_1[i], x \oplus K_2[i]] \leftarrow\!\!\$\ \overline{\mathrm{im}\, U[K_1[i], \cdot] \cup \mathrm{im}\, T[K_1[i], \cdot]}}$

  Else

    $U[K_1[i], x \oplus K_2[i]] \leftarrow T[K_1[i], x \oplus K_2[i]]$; $\mathsf{bad} \leftarrow \mathsf{true}$

    $\boxed{U[K_1[i], x \oplus K_2[i]] \leftarrow\!\!\$\ \overline{\mathrm{im}\, U[K_1[i], \cdot]}}$

Return $U[K_1[i], x \oplus K_2[i]] \oplus K_2[i]$

$\underline{\mathrm{E}(L, x)}$

If $T[L, x] = \bot$ then

  If $U[L, x] = \bot$ then

    $T[L, x] \leftarrow\!\!\$\ \overline{\mathrm{im}\, T[L, \cdot]}$

    If $T[L, x] \in U[L, \cdot]$ then

      $\mathsf{bad} \leftarrow \mathsf{true}$; $\boxed{T[L, x] \leftarrow\!\!\$\ \overline{\mathrm{im}\, T[L, \cdot] \cup \mathrm{im}\, U[L, \cdot]}}$

  Else $T[L, x] \leftarrow U[L, x]$; $\mathsf{bad} \leftarrow \mathsf{true}$; $\boxed{T[L, x] \leftarrow\!\!\$\ \overline{\mathrm{im}\, T[L, \cdot]}}$

Return $T[L, x]$

$\underline{\mathrm{E}^{-1}(L, y)}$

If $T^{-1}[L, y] = \bot$ then

  If $U^{-1}[L, y] = \bot$ then

  $x \leftarrow\!\!\$\ \overline{\mathrm{supp}\, T[L, \cdot]}$

    If $x \in \mathrm{supp}\, U[L, \cdot]$ then

      $\mathsf{bad} \leftarrow \mathsf{true}$; $\boxed{x \leftarrow\!\!\$\ \overline{\mathrm{supp}\, T[L, \cdot] \cup \mathrm{supp}\, U[L, \cdot]}}$

  Else $x \leftarrow U^{-1}[L, y]$; $\mathsf{bad} \leftarrow \mathsf{true}$; $\boxed{x \leftarrow\!\!\$\ \overline{\mathrm{supp}\, T[L, \cdot]}}$

  $T[L, x] \leftarrow y$

Return $T^{-1}[L, y]$

Figure 14: Modification of the sampling algorithm. In G_0, the values are sampled to keep consistency between $U[\cdot, \cdot]$ and $T[\cdot, \cdot]$, with the flag bad set if attempted independent sampling leads to inconsistencies. In G_1, the maps $U[\cdot, \cdot]$ and $T[\cdot, \cdot]$ are sampled independently, making RF an independent ideal cipher.

Theorem 5.7 *Let $\kappa, \lambda, \nu \geq 1$ be such that $\nu \leq \lambda - 2$. Let $\mathsf{H}\colon \{0,1\}^\lambda \times (\{0,1\}^* \times \{0,1\}^*) \to \{0,1\}^\lambda$ be a family of functions. Let $\mathsf{XCAU} = \mathbf{XCAU}[\mathsf{H}, \kappa, \lambda, \nu]$. Let A be an adversary that makes at most u queries to its* New *oracle, q_e queries per user instance to its* Enc *oracle with messages of length at most ℓ_{bit} bits, q_v queries per user instance to its* Vf *oracle with messages of length at most $\ell_{\text{bit}} + \lambda$ bits, and p queries to its* E *and* E^{-1} *oracles. Assume furthermore that $q_e \leq 2^\nu$, and $\ell_{\text{bit}} \leq \lambda(2^{\lambda-\nu} - 2)$. Then, with $\ell_{\text{blk}} = \lceil \ell_{\text{bit}}/\lambda \rceil + 1$,*

$$\mathsf{Adv}^{\text{mu-kr}}_{\mathsf{XCAU}}(A) \leq \frac{up(\ell_{\text{blk}}(q_e + q_v) + 1)}{2^{\lambda+\kappa+1}} + \frac{u}{2^\kappa}.$$

Proof: As in Theorem 5.1 and 5.3, we restrict our attention to adversaries A that do not use invalid user identifiers, that do not re-use nonces, and that do not verify ciphertexts obtained from the Enc oracle. The first step in this proof is to rewrite the game as G_0 in the same way as in the previous proofs; the scheme is changed to use the oracle RF that is, however, kept consistent with E and E^{-1}. The game is described in Fig. 15.

The next game G_1 is again a syntactic modification from G_0. We replace XCAU, which uses the unmodified block cipher and applies the input and output whitening as a part of the encryption and decryption procedures, by CAU instantiated with a block cipher with key length $\lambda+\kappa$. Consequently, we rewrite the oracle RF to perform the input and output whitening.

In the next game G_2, the oracles E and E^{-1}, and the oracle RF are based on different maps $T[\cdot,\cdot]$ (for E and E^{-1}) and $U[\cdot,\cdot]$ (for RF), but the oracles are defined to keep them consistent. This is achieved by first sampling them independently, but then re-sampling in case an inconsistency occurs. Should that be the case, the flag bad is set. Apart from this flag, games G_1 and G_2 are equivalent. We do not describe the game G_2 explicitly, but remark that it is obtained by verbatim replacement of the oracles E, E^{-1}, and RF in game G_1 by the ones described in game G_0 in Fig. 14. In the next game G_3, the re-sampling procedure keeping the oracles consistent is abandoned, which means that the oracles RF and E together with E^{-1} are independent. Like G_2, game G_3 is obtained by replacing the oracles E, E^{-1}, and RF by the ones in game G_1 in Fig. 14.

The probability of setting the bad flag in G_2 and G_3 can be bounded using Theorem 5.6. More technically, we describe an adversary $B = B(A)$ that emulates oracles to A as follows: Queries New, E, and E^{-1} by B are responded by B performing the same query in its game. Queries Enc and Dec are responded by B emulating the respective oracles using the oracle RF in its game to evaluate $\mathsf{CAU.Enc}$ and $\mathsf{CAU.Dec}$. The view of A is the same in G_2 and in the game $\mathrm{R}(B(A))$, and in G_3 and the game $\mathrm{S}(B(A))$, respectively. The numbers of queries u to the New oracle and p to the E and E^{-1} oracles are preserved by B. At most q_e queries of length at most ℓ_{bit} to Enc and at most q_v queries of length at most $\ell_{\text{bit}} + \lambda$ to Vf translate into at most $\ell_{\text{blk}}(q_e + q_v) + 1$ queries to RF in the game played by B. Using Theorem 5.6, this means that the probability of setting bad can be bounded by $up(\ell_{\text{blk}}(q_e + q_v) + 1)/2^{\lambda+\kappa+1}$.

All that remains to be done is bounding the probability of A guessing any key in G_3. As in this game, similarly to the previous proofs, the keys used to reference values in $U[\cdot,\cdot]$ is only used as an index to the table and is unrelated to all values that A observes in the game, the guessing probability is at most $u/2^\kappa$. This concludes the proof. ∎

6 Indistinguishability Security

In this section we prove the multi-user indistinguishability security bounds for CAU, RCAU, and XCAU, all in the ideal cipher model.

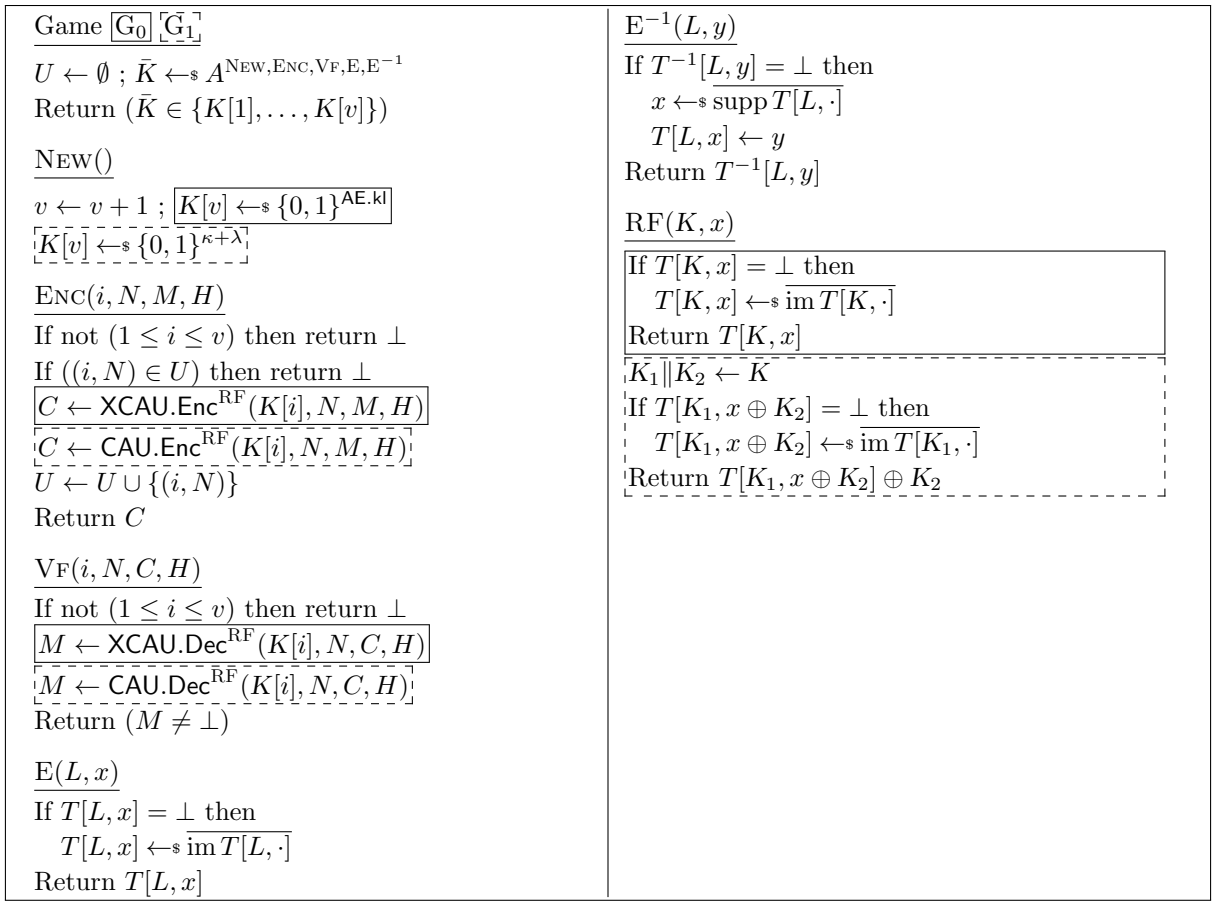

Figure 15: Games that intuitively correspond to the security of AES-XCAU (G_0) as well as AESX-CAU (G_1).

6.1 Preparation: A Lemma on CAU

We begin with a multi-user analysis of CAU which models the block cipher as a uniform random permutation and is useful in the subsequent proofs. The analysis is related to the ones of MV [24], IOM [18], and NOMI [27], with the main difference that they proved single-user security, while we directly prove multi-user security, and that our analysis is restricted to fixed-length nonces. We formalize the random-permutation model using our game $\mathbf{G}^{\text{mu-ind}}_{\mathrm{CAU}}$ while considering only adversaries that do not make use of the oracles E and E^{-1}.

Lemma 6.1 *Let $\kappa, \lambda, \nu \ge 1$ be such that $\nu \le \lambda - 2$. Let $\mathsf{H}\colon \{0,1\}^{\lambda} \times (\{0,1\}^* \times \{0,1\}^*) \to \{0,1\}^{\lambda}$ be an ϵ-almost XOR-universal hash function, for some $\epsilon\colon \mathbb{N} \times \mathbb{N} \to [0,1]$. Let $\mathsf{CAU} = \mathbf{CAU}[\mathsf{H}, \kappa, \lambda, \nu]$. Let A be an adversary that makes at most u queries to its* New *oracle, q_e queries per user instance to its* Enc *oracle with messages of length at most ℓ_{bit} bits, and q_v queries per user instance to its* Vf *oracle with messages of length at most $\ell_{\mathrm{bit}} + \lambda$ bits.[2] In particular, A does not use the* E *and* E^{-1} *oracles. Assume furthermore that $q_e \le 2^{\nu}$ and $\ell_{\mathrm{bit}} \le \lambda(2^{\lambda-\nu} - 2)$. Then*

$$\mathsf{Adv}^{\text{mu-ind}}_{\mathsf{CAU}}(A) \le \frac{u(u-1)}{2^{\kappa+1}} + \frac{u(\ell_{\mathrm{blk}}(q_e + q_v) + 1)^2 \cdot}{2^{\lambda+1}} + uq_v \cdot \left(2^{-\lambda} + \epsilon(\ell_{\mathrm{bit}}, \ell_{\mathrm{head}})\right),$$

[2]The ciphertext contains an λ-bit MAC tag, so the length of the contained plaintext is ℓ_{bit} bits.

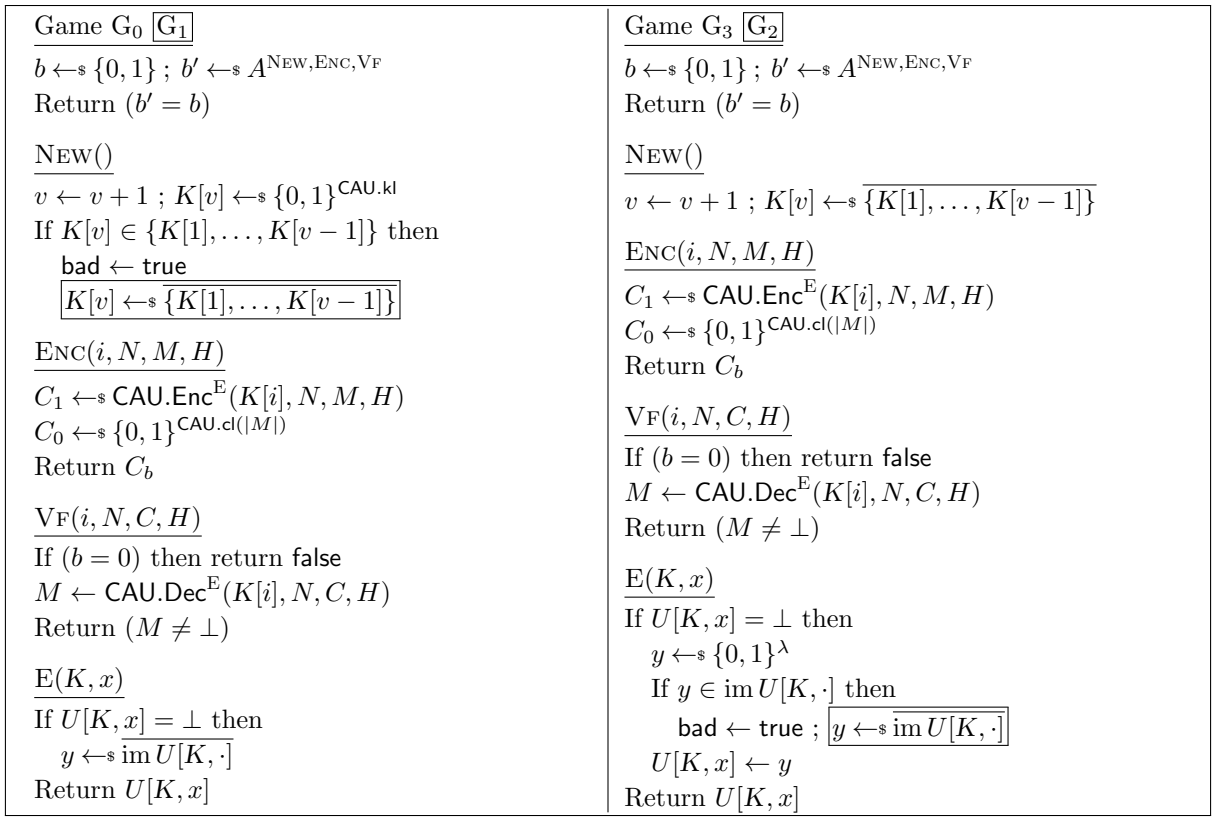

Figure 16: **Left:** Changing the game to prevent collisions among user keys. **Right:** Using a random function instead of a random permutation.

---

for $\ell_{\mathrm{blk}} = \lceil \ell_{\mathrm{bit}}/\lambda \rceil + 1$ *and where the AEAD headers are restricted to* ℓ_{head} *bits.*

Proof: We make the same assumptions about the validity of the queries made by adversary A as in Theorem 5.1. We describe a game G_0 in Fig. 16 that deviates from game $\mathbf{G}^{\text{mu-ind}}_{\mathsf{AE}}(A)$ in the New oracle, where we introduce a new flag bad that is set if a collision among the keys of the honest parties occurs. Game G_1 is almost the same, but we re-sample the key from the set of so-far unused keys if a freshly sampled key collides with one that was sampled before. The probability of bad to be set can easily be bounded by the collision probability $u(u-1)/2^{\kappa+1}$.

In game G_2 we first simplify the description of the oracle New in comparison with G_1. The next proof step is instrumental for establishing the secrecy of the scheme, the goal is to replace the ideal cipher used in CAU by an ideal random function with the same range and domain. This will allow us later to prove that the ciphertexts are uniformly distributed. We rewrite the game G_1 to the form of game G_2. The sampling procedure of the value $U[K,x]$ in E is an equivalent formulation that will allow the next proof step. The idea is, similar to proofs in [6], to first sample the output of the ideal cipher uniformly at random. In case a collision with previously sampled values occurrs, we sample uniformly from the set not containing the previous values. This does not change the distribution, since, intuitively, the probability mass associated to invalid responses is distributed uniformly over the valid responses. We additionally introduce a new flag bad that does not change the behavior of the game but detects when the re-sampling strategy is invoked.

Game G_4 $\boxed{\mathrm{G}_5}$	$\underline{\textsc{Vf}(i,N,C,H)}$
$b \leftarrow\!\!\$\ \{0,1\}$; $b' \leftarrow\!\!\$\ A^{\textsc{New},\textsc{Enc},\textsc{Vf}}$	If $(b=0)$ then return false
Return $(b'=b)$	$M \leftarrow \mathsf{CAU.Dec}^{\mathrm{E}}(K[i],N,C,H)$
$\underline{\textsc{New}()}$	If $M \neq \perp$ and $(i,N) \notin V$ then
$v \leftarrow v+1$; $K[v] \leftarrow\!\!\$\ \overline{\{K[1],\ldots,K[v-1]\}}$	$\mathsf{bad} \leftarrow \mathsf{true}$; $\boxed{\text{return false}}$
$\underline{\textsc{Enc}(i,N,M,H)}$	Return $(M \neq \perp)$
$V \leftarrow V \cup \{(i,N)\}$	$\underline{\mathrm{E}(K,x)}$
$C_1 \leftarrow\!\!\$\ \mathsf{CAU.Enc}^{\mathrm{E}}(K[i],N,M,H)$	If $U[K,x]=\perp$ then
$C_0 \leftarrow\!\!\$\ \{0,1\}^{\mathsf{CAU.cl}(\lvert M\rvert)}$	$U[K,x] \leftarrow\!\!\$\ \{0,1\}^{\lambda}$
Return C_b	Return $U[K,x]$

Figure 17: Between the games G_4 and G_5, we change the behavior of the $\textsc{Vf}$ oracle to reject forgery attempts also for $b=1$.

The next step is then to switch to game G_3 in which the oracle E samples uniform outputs. The games G_2 and G_3 differ only for after bad has been set, and the Fundamental Lemma then allows to bound the difference in advantage by the probability of provoking bad, where each block sampled initially in E is uniformly random. Since the collisions are checked for each key $K \in \{K[1],\ldots,K[u]\}$ individually, and the invocations of CAU lead to at most $\ell_{\mathrm{blk}}(q_e+q_v)+1$ queries per user, the Union Bound and the standard collision probability let us bound the overall probability by $u \cdot (\ell_{\mathrm{blk}}(q_e+q_v)+1)^2/2^{\lambda+1}$.

In game G_4 in Fig. 17, we introduce a new flag bad in the verification oracle $\textsc{Vf}$. The flag is set if the decryption algorithm $\mathsf{CAU.Dec}$ returns a valid result on a pair (i,N) of instance and nonce that has not been used in a query to $\textsc{Enc}$. In game G_5, also described in Fig. 17, we then modify what happens after the flag has been set: the oracle rejects anyway. The transition from G_4 to G_5 can be done in a sequence of hybrids $\mathrm{H}_0,\ldots,\mathrm{H}_{uq_v}$ where for queries to $\textsc{Vf}$ (with values (i,N) not used in prior queries to $\textsc{Enc}$) the game H_i returns false for the first i queries and $M \neq \perp$ thereafter. As $\mathrm{H}_0 = \mathrm{G}_4$ and $\mathrm{H}_{uq_v} = \mathrm{G}_5$, and $|\Pr[\mathrm{H}_i] - \Pr[\mathrm{H}_{i+1}]| \leq 2^{-\lambda}$, we conclude that $|\Pr[\mathrm{G}_4] - \Pr[\mathrm{G}_5]| \leq \frac{uq_v}{2^{\lambda}}$.

Game G_6 then changes how the ciphertexts in $\textsc{Enc}$ for $b=1$ are sampled. Instead of computing the tags as the xor of $\mathsf{H}(G,H,C)$ and $\mathrm{E}(K[i],Y+0)$, they are defined as $\mathrm{E}(K[i],Y+0)$. (One can view this as using the virtual block-cipher value $\mathrm{E}'(K[i],Y+0) = \mathrm{E}(K[i],Y+0) \oplus \mathsf{H}(G,H,C)$. As the adversary does not re-use nonces per instance, either value is uniformly random and independent of all other values.) To make the games consistent, we additionally have to modify $\textsc{Vf}$. For pairs (i,N) that have not been queried to $\textsc{Enc}$ previously, oracle $\textsc{Vf}$ always returns false. For pairs (i,N) that have been queried to $\textsc{Enc}$ previously, we change the condition so that $T \oplus \mathsf{H}(G,H,C) = \mathrm{E}'(K[i],Y+0) = \mathrm{E}(K[i],Y+0) \oplus \mathsf{H}(G,H',C')$ for H' and C' as used in the query to $\textsc{Enc}$.

In game G_6, we additionally introduce a flag bad that is set when the adversary successfully forges a tag, and G_7 differs from G_6 by always returning false even in case of a successful forgery. (This only changes the game after bad is set.) The difference in adversary advantage between games G_6 and G_7 can again be bounded by a sequence of hybrid games as above. The difference between each adjacent pair of games is then bounded by the probability that $\mathsf{H}(G,H',C') \oplus \mathsf{H}(G,H,C) = T \oplus \mathrm{E}(K[i],Y+0)$. Recall that H is an ϵ-almost XOR-universal hash function, and as the critical pair of evaluations of H is the first one during the game, we can bound the probability by

$$\Pr\left[\mathsf{H}(G,H,C) \oplus \mathsf{H}(G,H',C') = T \oplus \mathrm{E}(K[i],Y+0)\right] \leq \epsilon(\ell_{\mathrm{bit}},\ell_{\mathrm{head}}),$$

Game G_6 $\boxed{\mathrm{G}_7}$	$\textsc{Vf}(i, N, T\|\|C, H)$
$b \leftarrow\!\!\$\ \{0,1\}$; $b' \leftarrow\!\!\$\ A^{\textsc{New},\textsc{Enc},\textsc{Vf}}$	If $(b = 0$ or $(i, N) \notin V)$ then return false
Return $(b' = b)$	$G \leftarrow \mathrm{E}(K[i], 0^\lambda)$; $Y \leftarrow N\|\|\langle 1\rangle$
$\underline{\textsc{New}()}$	Let C', H' such that $(i, N, C', H') \in W$
$v \leftarrow v + 1$; $K[v] \leftarrow\!\!\$\ \overline{\{K[1], \ldots, K[v-1]\}}$	$\Delta \leftarrow T \oplus \mathrm{E}(K[i], Y + 0)$
$\underline{\textsc{Enc}(i, N, M, H)}$	If $\mathsf{H}(G, H', C') \oplus \mathsf{H}(G, H, C) = \Delta$ then
$G \leftarrow \mathrm{E}(K[i], 0^\lambda)$; $Y \leftarrow N\|\|\langle 1\rangle$	$\quad \mathsf{bad} \leftarrow \mathsf{true}$; $\boxed{\text{return false}}$
// Compute C as in $\mathsf{CAU.Enc}^{\mathrm{E}}(K[i], N, M, H)$	Return $\mathsf{H}(G, H', C') \oplus \mathsf{H}(G, H, C) = \Delta$
$C_1 \leftarrow \mathrm{E}(K[i], Y + 0)\|\|C$	
$V \leftarrow V \cup \{(i, N)\}$; $W \leftarrow W \cup \{(i, N, C, H)\}$	$\underline{\mathrm{E}(K, x)}$
$C_0 \leftarrow\!\!\$\ \{0,1\}^{\mathsf{CAU.cl}(\|M\|)}$	If $U[K, x] = \perp$ then
Return C_b	$\quad U[K, x] \leftarrow\!\!\$\ \{0,1\}^\lambda$
	Return $U[K, x]$

Figure 18: Game G_6 is equivalent to G_5. The outputs of $\textsc{Enc}$ are sampled differently, but $\textsc{Vf}$ is adapted in a consistent way.

---

by the fact that H is ϵ-almost XOR-universal. Adversary A makes at most uq_v queries to the $\textsc{Vf}$ oracle, which means that we can bound the overall probability that bad is set by $uq_v \cdot \epsilon(\ell_{\mathrm{bit}}, \ell_{\mathrm{head}})$.

Given that A makes at most $q_e \leq 2^\nu$ queries to $\textsc{Enc}$ per user instance, and the length of each message is bounded by $\ell_{\mathrm{blk}} \leq 2^{\lambda-\nu} - 2$ blocks, and because by assumption A does not repeat nonces, then in game G_7, the ciphertexts C_0 and C_1 are both uniformly random bit strings of the same length. The game is therefore independent of the challenge bit b, which means that the adversary cannot have any advantage in guessing b.

We combine all bounds shown in the above paragraphs:

$$\begin{aligned}
\mathsf{Adv}^{\text{mu-ind}}_{\mathsf{CAU}}(A) &= 2\Pr[\mathbf{G}^{\text{mu-ind}}_{\mathsf{CAU}}(A)] - 1 = 2\Pr[\mathrm{G}_0] - 1 \\
&\leq 2\Pr[\mathrm{G}_1] - 1 + \frac{u(u-1)}{2^{\kappa+1}} \\
&\leq 2\Pr[\mathrm{G}_3] - 1 + \frac{u(u-1)}{2^{\kappa+1}} + \frac{u((q_e+q_v)\cdot\ell_{\mathrm{blk}})^2}{2^{\lambda+1}} \\
&\leq 2\Pr[\mathrm{G}_5] - 1 + \frac{u(u-1)}{2^{\kappa+1}} + \frac{u((q_e+q_v)\cdot\ell_{\mathrm{blk}})^2}{2^{\lambda+1}} + uq_v \cdot 2^{-\lambda} \\
&\leq 2\Pr[\mathrm{G}_7] - 1 + \frac{u(u-1)}{2^{\kappa+1}} + \frac{u((q_e+q_v)\cdot\ell_{\mathrm{blk}})^2}{2^{\lambda+1}} + uq_v \cdot (2^{-\lambda} + \epsilon(\ell_{\mathrm{bit}}, \ell_{\mathrm{head}}))\ ,
\end{aligned}$$

which concludes the proof. ∎

6.2 Security of CAU

We now prove the multi-user indistinguishability security of plain CAU in the ideal-cipher model.

Theorem 6.2 *Let $\kappa, \lambda, \nu \geq 1$ be such that $\nu \leq \lambda - 2$. Let $\mathsf{H}\colon \{0,1\}^\lambda \times (\{0,1\}^* \times \{0,1\}^*) \to \{0,1\}^\lambda$ be an ϵ-almost XOR-universal hash function, for some $\epsilon\colon \mathbb{N} \times \mathbb{N} \to [0,1]$. Let $\mathsf{CAU} = \mathbf{CAU}[\mathsf{H}, \kappa, \lambda, \nu]$. Let A be an adversary that makes at most u queries to its $\textsc{New}$ oracle, q_e queries per user instance to its $\textsc{Enc}$ oracle with messages of length at most ℓ_{bit} bits, q_v queries per user instance to its $\textsc{Vf}$*

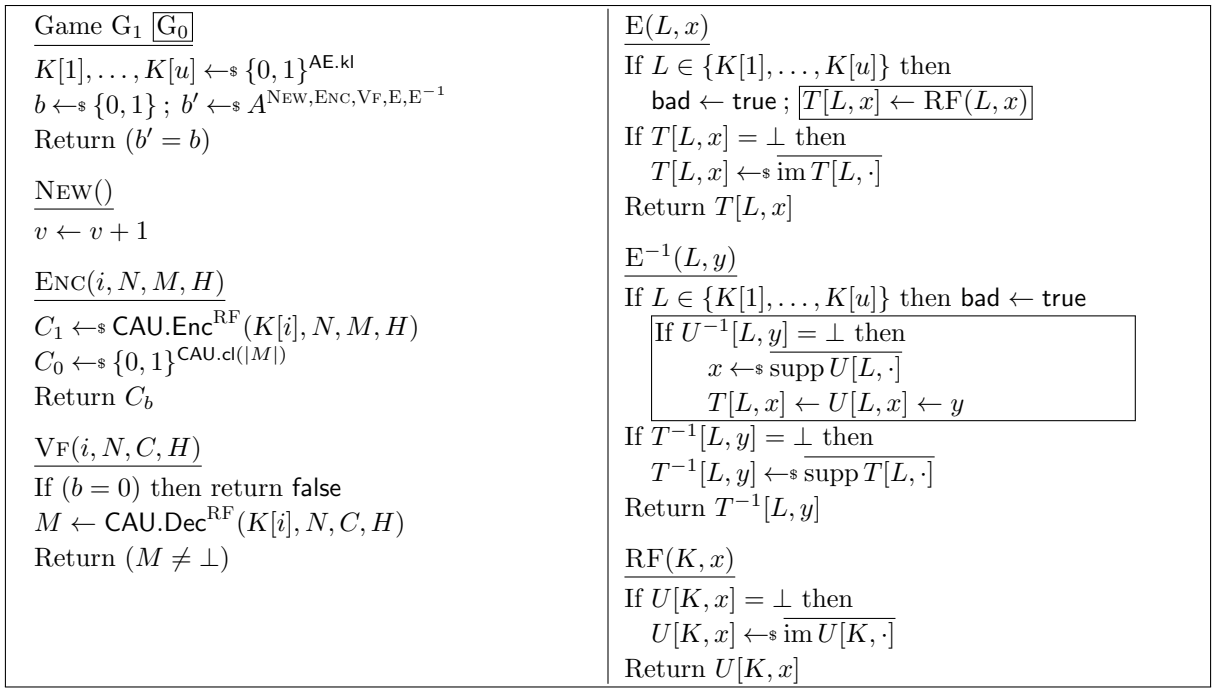

Figure 19: In game G_0 the oracles RF, E, and E^{-1} are all consistent, in game G_1, oracle RF is sampled independently.

---

oracle with messages of length at most $\ell_{\mathrm{bit}} + \lambda$ bits, and p queries to its E *and* E^{-1} *oracles. Assume furthermore that $q_e \leq 2^{\nu}$ and $\ell_{\mathrm{bit}} \leq \lambda(2^{\lambda-\nu} - 2)$. Then*

$$\mathsf{Adv}^{\text{mu-ind}}_{\mathsf{CAU}}(A) \leq \frac{up}{2^{\kappa}} + \frac{u(\ell_{\mathrm{blk}}(q_e + q_v) + 1)^2 \cdot}{2^{\lambda+1}} + \frac{u(u-1)}{2^{\kappa+1}} + uq_v \cdot \left(2^{-\lambda} + \epsilon(\ell_{\mathrm{bit}}, \ell_{\mathrm{head}})\right),$$

for $\ell_{\mathrm{blk}} = \lceil \ell_{\mathrm{bit}}/\lambda \rceil + 1$ and where the AEAD headers are restricted to ℓ_{head} bits.

The first term originates from the advantage of the adversary in guessing a user's key in a query to the ideal cipher. This term grows linearly in the number of honest sessions, and it also grows linearly in the number of adversary calls to the ideal cipher. The second term stems from a PRF/PRP-switching in the proof of counter mode. The third term stems from a potential collision of honest-user keys, and the final term from the authentication using the AUH-based MAC.

Proof: We make the same assumptions about the validity of the queries made by adversary A as in Theorem 5.1. The first game G_0 we consider is described in Fig. 19 and is only a syntactic modification of game $\mathbf{G}^{\text{mu-ind}}_{\mathsf{CAU}}(A)$. We introduce an additional oracle RF that implements the forward evaluation of the ideal cipher for the algorithms $\mathsf{CAU.Enc}$ and $\mathsf{CAU.Dec}$. This is sufficient as encryption and decryption in GCM never query E^{-1}. In more detail, oracle RF samples the ideal cipher for the keys used in the encryption using the map $U[\cdot, \cdot]$. The oracles E and E^{-1} are adapted such that, for keys used in the game, they sample the map $T[\cdot, \cdot]$ consistently with $U[\cdot, \cdot]$. We also change how keys for new user instances are sampled. Instead of sampling them within New, we sample all keys $K[1], \ldots K[u]$ in the beginning of the game. Finally, we introduce a flag bad that is set when the adversary A queries one of the oracles E or E^{-1} with a key that is also used by an honest user instance. Apart from this flag, game G_0 is equivalent to $\mathbf{G}^{\text{mu-ind}}_{\mathsf{CAU}}(A)$.

We then rewrite the game in the form of G_1 analogously to the modifications in Theorem 5.1,

which modifies the way in which the responses for the E, E^{-1}, and RF oracles are determined. In particular, we break the consistency between E and E^{-1} on the one hand, and RF on the other hand, by sampling the oracle responses independently. Since all changes appear only after bad has been set, we can relate the games using the Fundamental Lemma from Bellare and Rogaway [6] and proceed by bounding the probability of setting bad. This probability is bounded by $up/2^{\kappa}$. The proof step follows exactly as the one in Theorem 5.1, namely by describing for each adversary A a non-adaptive adversary $B = B(A)$ that achieves the same probability in provoking bad as A by emulating all oracles to A, recording all queries, and re-playing them adaptively in the game. We then follow the exact steps of Theorem 5.1 to conclude that the probability of provoking bad is bounded by $\frac{up}{2^{\kappa}}$.

We now observe that winning game G_1 is almost equivalent to winning $\mathbf{G}^{\text{mu-ind}}_{\mathsf{CAU}}$ without access to the oracles E and E^{-1}, but these oracles are independent of all other oracles in the game, and so they are easy to simulate. We therefore consider the adversary $C = C(A)$ in the game $\mathbf{G}^{\text{mu-ind}}_{\mathsf{CAU}}$ that simulates the oracles E and E^{-1} to the adversary A and forwards all other queries to its own oracles. Adversary C therefore never uses the oracles E and E^{-1}, therefore we can apply Lemma 6.1. Clearly, $\Pr[\mathrm{G}_1] = \Pr[\mathbf{G}^{\text{mu-ind}}_{\mathsf{CAU}}(C(A))]$, and therefore

$$\begin{aligned}
\mathsf{Adv}^{\text{mu-ind}}_{\mathsf{CAU}}(A) &= 2\Pr[\mathbf{G}^{\text{mu-ind}}_{\mathsf{CAU}}(A)] - 1 = 2\Pr[\mathrm{G}_0] - 1 \\
&\leq 2\Pr[\mathbf{G}^{\text{mu-ind}}_{\mathsf{CAU}}(C(A))] - 1 + \frac{up}{2^{\kappa}} \\
&\leq \frac{u(u-1)}{2^{\kappa+1}} + \frac{u((q_e+q_v)\cdot\ell_{\text{blk}})^2}{2^{\lambda+1}} + uq_v \cdot \left(2^{-\lambda} + \epsilon(\ell_{\text{bit}}, \ell_{\text{head}})\right) + \frac{up}{2^{\kappa}},
\end{aligned}$$

which concludes the proof. ∎

6.3 Security of RCAU

In terms of bounds for RCAU, we first show a simple corollary proving that the same bounds as for CAU also apply for RCAU. This follows immediately by a reduction that randomizes the nonces, and shows that the randomization of the nonces does not degrade security.

Corollary 6.3 *Let $\kappa, \lambda, \nu \geq 1$ be such that $\nu \leq \lambda - 2$. Let $\mathsf{H}\colon \{0,1\}^{\lambda} \times (\{0,1\}^* \times \{0,1\}^*) \to \{0,1\}^{\lambda}$ be an ϵ-almost XOR-universal hash function, for some $\epsilon\colon \mathbb{N} \times \mathbb{N} \to [0,1]$. Let $\mathsf{RCAU} = \mathbf{RCAU}[\mathsf{H}, \kappa, \lambda, \nu]$. Let A be an adversary that makes at most u queries to its* New *oracle, q_e queries per user instance to its* Enc *oracle with messages of length at most ℓ_{bit} bits, q_v queries per user instance to its* Vf *oracle with messages of length $\ell_{\text{bit}} + \lambda$ bits, p_e queries to its* E *oracle, p_i queries to its E^{-1} oracle, and $p = p_e + p_i$. Assume furthermore that $q_e \leq 2^{\nu}$, and $\ell_{\text{bit}} \leq \lambda(2^{\lambda-\nu} - 2)$. (For brevity we write $q = q_e + q_v$.) Then*

$$\mathsf{Adv}^{\text{mu-ind}}_{\mathsf{RCAU}}(A) \leq \frac{up}{2^{\kappa}} + \frac{u(\ell_{\text{blk}}(q_e+q_v)+1)^2\cdot}{2^{\lambda+1}} + \frac{u(u-1)}{2^{\kappa+1}} + uq_v \cdot \left(2^{-\lambda} + \epsilon(\ell_{\text{bit}}, \ell_{\text{head}})\right), \qquad (8)$$

for $\ell_{\text{blk}} = \lceil \ell_{\text{bit}}/\lambda \rceil + 1$ and where the AEAD headers are restricted to ℓ_{head} bits.

Proof: We prove this via a reduction as follows: adversary B within game $\mathbf{G}^{\text{mu-ind}}_{\mathsf{CAU}}$ emulates the oracles of game $\mathbf{G}^{\text{mu-ind}}_{\mathsf{RCAU}}$ toward the assumed adversary A as follows: queries to oracles E and E^{-1} are simply forwarded to the same oracles in game $\mathbf{G}^{\text{mu-ind}}_{\mathsf{CAU}}$. Queries to oracle New are handled by calling the same oracle in $\mathbf{G}^{\text{mu-ind}}_{\mathsf{CAU}}$, and for the ith user sampling a random masking key $K_2[i] \leftarrow\!\!\$\ \{0,1\}^{\nu}$. Queries $\textsc{Enc}(i, N, M, H)$ and $\textsc{Dec}(i, N, C, H)$ are handled by calling the respective oracle in $\mathbf{G}^{\text{mu-ind}}_{\mathsf{CAU}}$, but replacing N by $N \oplus K_2[i]$. The view of A in $\mathbf{G}^{\text{mu-ind}}_{\mathsf{RCAU}}(A)$ and in

$\mathbf{G}^{\text{mu-ind}}_{\mathsf{CAU}}(B(A))$ is the same. As B makes exactly the same number of queries as A, the claimed bound follows. ▌

We prove a stronger bound for the advantage of adversaries that do not use its VF oracle. The bound differs from the one proven above significantly: we show that we can replace the term $up/2^\kappa$ in the bound for CAU by terms that are smaller for realistic parameters. The proof does, however, not immediately apply to adversaries that *do* make use of the VF oracle. In the result for RCAU, we explicitly distinguish between the numbers for evaluation p_e and inversion p_i queries for the block cipher, with $p = p_e + p_i$.

Theorem 6.4 *Let $\kappa, \lambda, \nu \geq 1$ be such that $\nu \leq \lambda - 2$. Let $\mathsf{H}\colon \{0,1\}^\lambda \times (\{0,1\}^* \times \{0,1\}^*) \to \{0,1\}^\lambda$ be a family of functions. Let $\mathsf{RCAU} = \mathbf{RCAU}[\mathsf{H}, \kappa, \lambda, \nu]$. Let A be an adversary that makes at most u queries to its* NEW *oracle, q_e queries per user instance to its* ENC *oracle with messages of length at most ℓ_{bit} bits, no queries to its* VF *oracle, p_e queries to its* E *oracle, p_i queries to its* E^{-1} *oracle, and $p = p_e + p_i$. Assume furthermore that $q_e \leq 2^\nu$, and $\ell_{\text{bit}} \leq \lambda(2^{\lambda-\nu} - 2)$. Then*

$$\mathsf{Adv}^{\text{mu-ind}}_{\mathsf{RCAU}}(A) \leq \frac{u(u-1)}{2^{\kappa+1}} + \frac{(c+1)\gamma \cdot up(q_e\ell_{\text{blk}} + 1)}{(2^\kappa - p)(2^\lambda - p)} + \frac{2\gamma \cdot upq_e}{(2^\kappa - p)(2^\nu - pq_e)} + \frac{c\gamma \cdot p_i}{2^\kappa - p}$$
$$+ \frac{\gamma \cdot upq_e \cdot \ell_{\text{blk}}}{(2^\kappa - p)(2^\lambda - (q_e\ell_{\text{blk}} + 1))} + (p + uq_e\ell_{\text{blk}}) \left(\frac{1}{2^\lambda - q_e\ell_{\text{blk}}}\right)^c \binom{uq_e(\ell_{\text{blk}} + 1)}{c+1} + \frac{u(q_e\ell_{\text{blk}})^2}{2^{\lambda+1}}\ , \quad (9)$$

for $\ell_{\text{blk}} = \lceil \ell_{\text{bit}}/\lambda \rceil + 1$, for any constant $c \in \mathbb{N}$, and for $\gamma = 2^\kappa/(2^\kappa - pq_e)$.

We again start by providing a corollary that reformulates the bound under reasonable assumptions on the numbers of queries.

Corollary 6.5 *Let all variables be as defined in Theorem 6.4, and assume that $p \leq 2^{\kappa-1}$, $p, q_e\ell_{\text{blk}} + 1 \leq 2^{\lambda-1}$, and $pq_e \leq 2^{\nu-1}$. Then*

$$\mathsf{Adv}^{\text{mu-ind}}_{\mathsf{RCAU}}(A) \leq \frac{u(u-1)}{2^{\kappa+1}} + \frac{(c+2)\gamma \cdot up(q_e\ell_{\text{blk}} + 1)}{2^{\kappa+\lambda-2}} + \frac{2\gamma \cdot upq_e}{2^{\kappa+\nu-2}} + \frac{c\gamma \cdot p_i}{2^{\kappa-1}}$$
$$+ (p + uq_e\ell_{\text{blk}}) \left(\frac{1}{2^\lambda - q_e\ell_{\text{blk}}}\right)^c \binom{uq_e(\ell_{\text{blk}} + 1)}{c+1} + \frac{u(q_e\ell_{\text{blk}})^2}{2^{\lambda+1}}\ . \quad (10)$$

In comparison with the bound proven in Theorem 6.2, the major difference in Equation (10) is that the term $up/2^\kappa$ is replaced by multiple terms:

- terms in which the denominator grows exponentially in two parameters, namely $(c+2)\gamma \cdot upq_e \cdot (\ell_{\text{blk}} + 1)$ / $2^{\kappa+\lambda-2}$ and $2\gamma \cdot upq_e$ / $2^{\kappa+\nu-2}$, and which are therefore significantly smaller,
- the multi-collision term that can be made small by choosing moderate values of c, and
- the term $c\gamma \cdot p_i$ / $2^{\kappa-1}$ that does not scale with the number of users.

Our theorem does not support a similar improvement for adversaries that make queries to their VF oracle.

Proof of Theorem 6.4: The proof begins along the same lines as the proof of Theorem 5.3 and continues along the lines of Theorem 6.2. Therefore, we mention many arguments in the proof only briefly and refer to the more detailed discussions in the other proofs. We again restrict our attention to adversaries A that do not use invalid user identifiers and do not re-use nonces, and

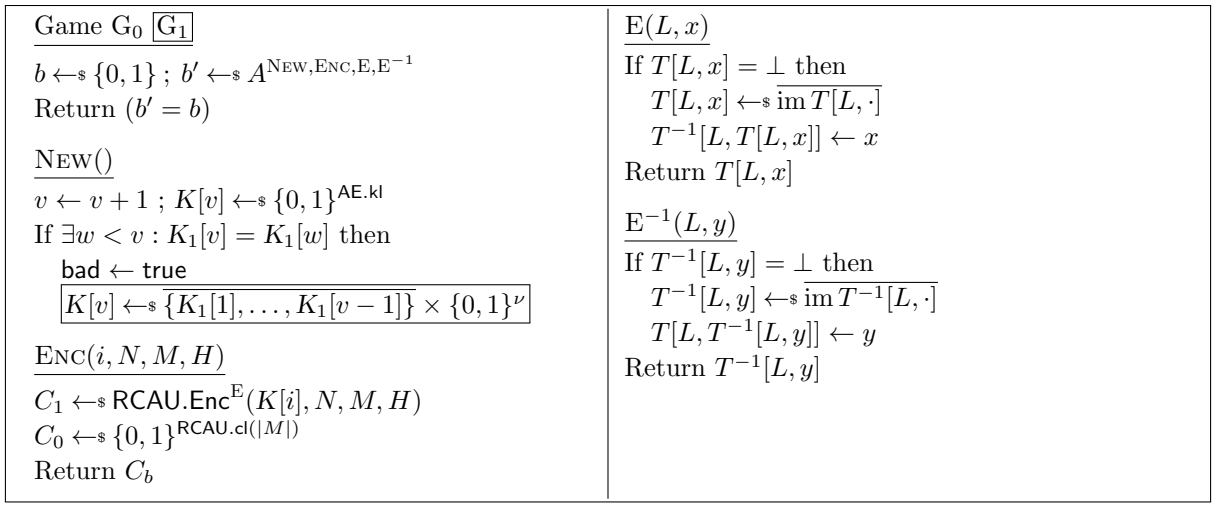

Figure 20: Game G_0 sets bad in case a collision occurs between the block-cipher keys of honest user instances. Game G_1 samples the keys so that no such collision occurs.

similarly to the previous proofs the strategy here is based on decoupling the oracles E and E^{-1} available to the adversary from the block cipher used in RCAU.Enc.

In game G_0, described in detail in Fig. 20, the difference to $\mathbf{G}^{\text{mu-ind}}_{\mathsf{RCAU}}(A)$ is that we add the flag bad and set it if, for a new user instance, the part of the key that is used as a the key of the block cipher coincides with the corresponding part of a key of an existing user instance. As in the proof of Theorem 5.3, for a user instance $i \in \{1, \ldots, v\}$, the key consists of two parts $K[i] = K_1[i] \| K_2[i]$ with $K_1[i] \in \{0,1\}^{\kappa}$ and $K_2[i] \in \{0,1\}^{\nu}$, where $K_1[i]$ is used as a key to the block cipher and $K_2[i]$ is used to mask the input. In the subsequent game G_1, we additionally re-sample the key so that such a collision does not occur. As G_0 and G_1 differ only after bad is set, we can bound the difference of the adversary's advantage by the probability of bad being set. Since at most u user instances are created, that probability can be bounded by $u(u-1)/2^{\kappa+1}$. In the subsequent steps, we can therefore sample the keys as in G_1, namely without collisions in the first part.

The next step, as in Theorem 5.3, introduces an additional oracle RF that is accessed by RCAU.Enc in Enc instead of E, as in previous proofs. The oracles are re-defined to keep the maps $T[\cdot, \cdot]$ and $U[\cdot, \cdot]$ synchronized. In contrast to $T[\cdot, \cdot]$, however, $U[\cdot, \cdot]$ only contains values corresponding to honest user instances and is therefore addressed by user indices instead of keys, i.e., $U[i, \cdot]$ as opposed to $T[K_1[i], \cdot]$. We apply this change also to the encryption algorithm and the oracle Enc, where (by slight abuse of notation) we generally use the user index i instead of the key $K[i]$. Another modification changes where the block-cipher inputs are randomized; in game G_1 this occurs within the algorithm RCAU.Enc, in game G_2 we take the perspective that this becomes part of the oracle RF—and even the map $U[\cdot, \cdot]$. Consequently, we replace the encryption algorithm $\mathsf{RCAU.Enc}^{\text{E}}$ in Enc by the non-randomized variant $\mathsf{AE.Enc}^{\text{RF}}$. This is only a syntactic change. To summarize, in game G_1, the algorithm RCAU.Enc is called in oracle Enc with the key $K[i]$ for user i as parameter, algorithm RCAU.Enc then uses the first κ bits of $K[i]$ as key in calls to E and the last ν bits for randomizing the first ν bits of inputs to E (except for 0^{λ}, which is not randomized). In game G_2, the algorithm AE.Enc is called in oracle Enc with the user index i as parameter. Algorithm AE.Enc also uses the index i in calls to oracle RF, and oracle RF uses the appropriate key $K_1[i]$ as first argument to $U[\cdot, \cdot]$ and masks the first ν bits of input x with $K_2[i]$, as previously done within

Game $\boxed{\mathrm{G}_2}$ G_3

$b \leftarrow\!\!\$\ \{0,1\}$; $b' \leftarrow\!\!\$\ A^{\textsc{New},\textsc{Enc},\mathrm{E},\mathrm{E}^{-1}}$
Return $(b' = b)$

$\underline{\textsc{New}()}$
$v \leftarrow v + 1$
$K[v] \leftarrow\!\!\$\ \overline{\{K_1[1], \ldots, K_1[v-1]\}} \times \{0,1\}^\nu$

$\underline{\textsc{Enc}(i, N, M, H)}$
$C_1 \leftarrow\!\!\$\ \mathsf{AE.Enc}^{\mathrm{RF}}(i, N, M, H)$
$C_0 \leftarrow\!\!\$\ \{0,1\}^{\mathsf{AE.cl}(|M|)}$
Return C_b

$\underline{\mathrm{E}(L, x)}$
If $T[L, x] = \bot$ then
  $T[L, x] \leftarrow\!\!\$\ \overline{\mathrm{im}\, T[L, \cdot]}$
  If $\exists j : K_1[j] = L$ then
    If $x = 0^\lambda$ and $U[j, 0^\lambda] \neq \bot$ then
      $T[L, 0^\lambda] \leftarrow U[j, 0^\lambda]$
    If $x \neq 0^\lambda$ and $U[j, K_2[j] \oplus x] \neq \bot$ then
      $\mathsf{bad} \leftarrow \mathsf{true}$; $\boxed{T[L, x] \leftarrow U[j, K_2[j] \oplus x]}$
    If $T[L, x] \in \mathrm{im}\, U[j, \cdot]$ then
      $\mathsf{bad} \leftarrow \mathsf{true}$
      $\boxed{T[L, x] \leftarrow\!\!\$\ \overline{\mathrm{im}\, T[L, \cdot] \cup \mathrm{im}\, U[j, \cdot]}}$
Return $T[L, x]$

$\underline{\mathrm{E}^{-1}(L, y)}$
If $T^{-1}[L, y] = \bot$ then
  $x \leftarrow\!\!\$\ \overline{\mathrm{supp}\, T[L, \cdot]}$
  If $\exists j : K_1[j] = L$ then
    If $U[j, 0^\lambda] = y$ then $x \leftarrow 0^\lambda$
    If $y \in U[j, \overline{\{0^\lambda\}}]$ then
      $\mathsf{bad} \leftarrow \mathsf{true}$; $\boxed{x \leftarrow U^{-1}[j, y] \oplus K_2[j]}$
    If $x \in \mathrm{supp}\, U[j, \cdot]$ then
      $\mathsf{bad} \leftarrow \mathsf{true}$
      $\boxed{x \leftarrow\!\!\$\ \overline{\mathrm{supp}\, T[L, \cdot] \cup \mathrm{supp}\, U_\$[j, \cdot]}}$
  $T[L, x] \leftarrow y$
Return $T^{-1}[L, y]$

$\underline{\mathrm{RF}(i, x)}$
If $U[i, x] = \bot$ then
  $U[i, x] \leftarrow\!\!\$\ \overline{\mathrm{im}\, U[i, \cdot]}$
  If $x = 0^\lambda$ and $T[K_1[i], 0^\lambda] \neq \bot$ then
    $U[i, 0^\lambda] \leftarrow T[K_1[i], 0^\lambda]$
  If $x \neq 0^\lambda$ and $T[K_1[i], K_2[i] \oplus x] \neq \bot$ then
    $\mathsf{bad} \leftarrow \mathsf{true}$; $\boxed{U[i, x] \leftarrow T[K_1[i], K_2[i] \oplus x]}$
  If $U[i, x] \in \mathrm{im}\, T[K_1[i], \cdot]$ then
    $\mathsf{bad} \leftarrow \mathsf{true}$
    $\boxed{U[i, x] \leftarrow\!\!\$\ \overline{\mathrm{im}\, T[K_1[i], \cdot] \cup \mathrm{im}\, U[i, \cdot]}}$
Return $U[i, x]$

Figure 21: Game G_2 is almost the same as G_1 but changes the sampling procedure in E, E^{-1}, and RF, as well as the addressing of the keys in RF. Game G_3 differs in that the consistency between T and U is dropped.

RCAU.Enc. Here, and in the following, we write $K_2[i] \oplus x$ to mean that $K_2[i] \in \{0,1\}^\nu$ is xored to the first ν bits of $x \in \{0,1\}^\lambda$. The oracle RF also handles the input 0^λ appropriately, namely without randomization.

More technically, the outputs of the oracles E, E^{-1}, and RF are first sampled as if the maps $T[\cdot, \cdot]$ and $U[\cdot, \cdot]$ were independent; namely uniformly from $\overline{\mathrm{im}\, T[L, \cdot]}$ in E, from $\overline{\mathrm{supp}\, T[L, \cdot]}$ in E^{-1}, and from $\overline{\mathrm{im}\, U[i, \cdot]}$ in RF. Then, the oracles check whether the sampled value violates the equation $T[K_1[i], K_2[i] \oplus x] = U[i, x]$ for all $x \neq 0^\lambda$ that must be preserved. In that case, the response is set to the consistent value (if it is determined by the previous state) or re-sampled according to the condition (if it is not determined but the newly sampled value violates the condition). In all cases where a violation occurs, the flag bad is set. The value 0^λ is treated specially because it is the only value not randomized in RF, and therefore $T[K_1[i], 0^\lambda] = U[i, 0^\lambda]$ must be satisfied; the oracles assure that this condition is preserved but unlike for $x \neq 0^\lambda$ they do not set the flag bad if the value is copied from the respective other map. Overall, all changes from G_1 to G_2, which is explicitly described in Fig. 21, only change the game syntactically and therefore $\Pr[\mathrm{G}_1] = \Pr[\mathrm{G}_2]$.

In game G_3, which is also described in Fig. 21 and does not contain the boxed code, so the sampled outputs in the oracles E, E^{-1}, and RF, are not overwritten after bad is set. This is the only modification, and by the Fundamental Lemma of Game Playing, we obtain that $|\Pr[\mathrm{G}_2] - \Pr[\mathrm{G}_3]| \leq$

Game G_4 $\boxed{\mathrm{G}_5}$

$b \leftarrow\!\!\$\ \{0,1\}$; $b' \leftarrow\!\!\$\ A^{\textsc{New},\textsc{Enc},\mathrm{E},\mathrm{E}^{-1}}$
Return $(b' = b)$

$\underline{\textsc{New}()}$
$v \leftarrow v + 1$
$K[v] \leftarrow\!\!\$\ \overline{\{K_1[1], \ldots, K_1[v-1]\}} \times \{0,1\}^\nu$

$\underline{\textsc{Enc}(i, N, M, H)}$
$C_1 \leftarrow\!\!\$\ \mathsf{AE.Enc}^{\mathrm{RF}}(i, N, M, H)$
$C_0 \leftarrow\!\!\$\ \{0,1\}^{\mathsf{AE.cl}(|M|)}$
Return C_b

$\underline{\mathrm{E}(L, x)}$
If $T[L, x] = \bot$ then
  $T[L, x] \leftarrow\!\!\$\ \overline{\mathrm{im}\, T[L, \cdot]}$
  If $\exists j : K_1[j] = L$ then
    If $x = 0^\lambda$ and $U[j, 0^\lambda] \neq \bot$ then
      $T[L, 0^\lambda] \leftarrow U[j, 0^\lambda]$
    If ($x \neq 0^\lambda$ and $U[j, K_2[j] \oplus x] \neq \bot$)
      or $T[L, x] \in \mathrm{im}\, U[j, \cdot]$ then $\mathsf{bad} \leftarrow \mathsf{true}$
    $(\tilde{K}, \tilde{U}[\cdot, 0^\lambda]) \leftarrow\!\!\$\ \mathrm{GK}^v_{T,U}$
    $\tilde{U}[\cdot, \overline{\{0^\lambda\}}] \leftarrow U[\cdot, \overline{\{0^\lambda\}}]$; $\boxed{U[\cdot, 0^\lambda] \leftarrow \tilde{U}[\cdot, 0^\lambda]}$
    If $\exists y \in \{0,1\}^\lambda : |\tilde{U}^{-1}[\cdot, y]| > c$ then
      $\mathsf{bad} \leftarrow \mathsf{true}$
    $\boxed{K \leftarrow \tilde{K}}$
Return $T[L, x]$

$\underline{\mathrm{E}^{-1}(L, y)}$
If $T^{-1}[L, y] = \bot$ then
  $x \leftarrow\!\!\$\ \overline{\mathrm{supp}\, T[L, \cdot]}$
  If $\exists j : K_1[j] = L$ then
    If $U[j, 0^\lambda] = y$ then $x \leftarrow 0^\lambda$
    If $y \in U[j, \overline{\{0^\lambda\}}]$ or $x \in \mathrm{supp}\, U[j, \cdot]$ then
      $\mathsf{bad} \leftarrow \mathsf{true}$
    $(\tilde{K}, \tilde{U}[\cdot, 0^\lambda]) \leftarrow\!\!\$\ \mathrm{GK}^v_{T,U}$
    $\tilde{U}[\cdot, \overline{\{0^\lambda\}}] \leftarrow U[\cdot, \overline{\{0^\lambda\}}]$; $\boxed{U[\cdot, 0^\lambda] \leftarrow \tilde{U}[\cdot, 0^\lambda]}$
    If $\exists y \in \{0,1\}^\lambda : |\tilde{U}^{-1}[\cdot, y]| > c$ then
      $\mathsf{bad} \leftarrow \mathsf{true}$
    $\boxed{K \leftarrow \tilde{K}}$
  $T[L, x] \leftarrow y$
Return $T^{-1}[L, y]$

$\underline{\mathrm{RF}(i, x)}$
If $U[i, x] = \bot$ then
  $U[i, x] \leftarrow\!\!\$\ \overline{\mathrm{im}\, U[i, \cdot]}$
  If $x = 0^\lambda$ and $T[K_1[i], 0^\lambda] \neq \bot$ then
    $U[i, 0^\lambda] \leftarrow T[K_1[i], 0^\lambda]$
  If $x \neq 0^\lambda$ and $T[K_1[i], K_2[i] \oplus x] \neq \bot$ then
    $\mathsf{bad} \leftarrow \mathsf{true}$
  If $U[i, x] \in \mathrm{im}\, T[K_1[i], \cdot]$ then $\mathsf{bad} \leftarrow \mathsf{true}$
  $(\tilde{K}, \tilde{U}[\cdot, 0^\lambda]) \leftarrow\!\!\$\ \mathrm{GK}^v_{T,U}$
  $\tilde{U}[\cdot, \overline{\{0^\lambda\}}] \leftarrow U[\cdot, \overline{\{0^\lambda\}}]$; $\boxed{U[\cdot, 0^\lambda] \leftarrow \tilde{U}[\cdot, 0^\lambda]}$
  If $\exists y \in \{0,1\}^\lambda : |\tilde{U}^{-1}[\cdot, y]| > c$ then $\mathsf{bad} \leftarrow \mathsf{true}$
  $\boxed{K \leftarrow \tilde{K}}$
Return $U[i, x]$

Figure 22: Game G_4 re-samples the keys $K[i]$ and $U[i, 0^\lambda]$ and introduces a further condition on collisions of values in $U[\cdot, \cdot]$. Game G_5 makes the newly sampled keys persistent.

$\Pr[\mathsf{bad} \text{ in } \mathrm{G}_3]$.

In game G_4, described in Fig. 22, the description of the oracles E, E^{-1}, and RF is simplified. There is also a further change to these oracles. After the return value of the respective oracle has been sampled, the game samples keys $(\tilde{K}[1], \ldots, \tilde{K}[v])$ and block-cipher values $(\tilde{U}[1, 0^\lambda], \ldots, \tilde{U}[v, 0^\lambda])$ from the set $\mathrm{GK}^v_{T,U}$, as described in the proof of Theorem 5.3. The flag bad is then also set when, for some value $y \in \{0,1\}^\lambda$, more than c pairs of index and block $(i, x) \in \{1, \ldots, v\} \times \{0,1\}^\lambda$ satisfy $U[i, x] = y$ or $\tilde{U}[j, 0^\lambda] = y$. The exact purpose of this modification will become clear in subsequent proof steps; essentially, introducing this event will simplify bounding the overall probability of bad being set, since every output value in the map $T[\cdot, \cdot]$ that is defined in E or E^{-1} can affect at most $c \in \mathbb{N}$ values in $U[\cdot, \cdot]$.

As in Theorem 5.3, in terms of adversary advantage and in fact independent of the above-described distribution, game G_4 is still equivalent to G_3, the main difference between the two games being that the bad flag is more likely to be set in G_4. The adversary advantage in the two games, however, is unaffected, meaning that $\Pr[\mathrm{G}_4] = \Pr[\mathrm{G}_3]$ and $\Pr[\mathsf{bad} \text{ in } \mathrm{G}_3] \leq \Pr[\mathsf{bad} \text{ in } \mathrm{G}_4]$.

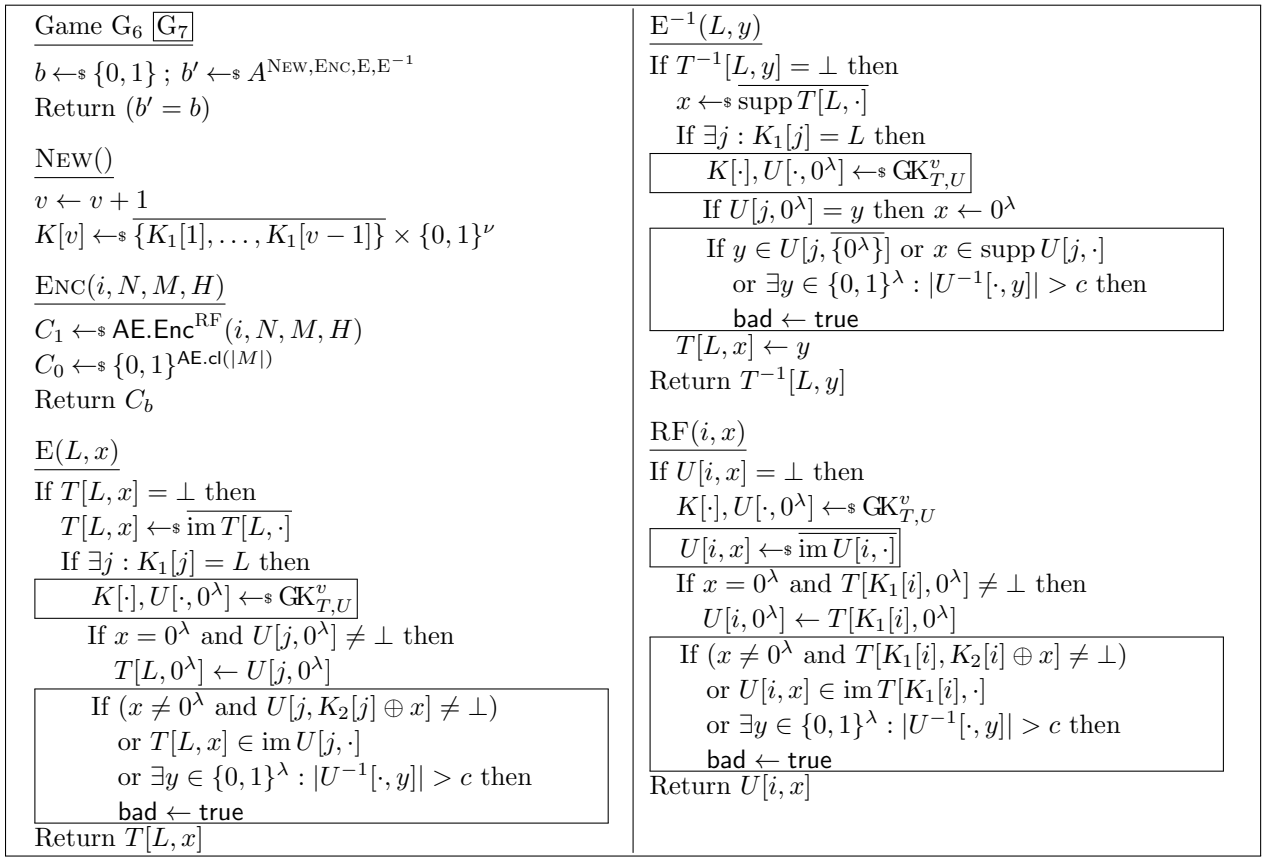

Figure 23: In G_6, the re-sampling is shifted to the beginning of the oracles.

In G_5, which is also described in Fig. 22, we then change the handling of the keys $K[\cdot]$. At the end of each oracle E, E^{-1}, and RF, the key vector $(K[1], \ldots, K[v])$ is set to the values $(\tilde{K}[1], \ldots, \tilde{K}[v])$ sampled from the set $\text{GK}^v_{T,U}$ of good keys. Exactly as in Theorem 5.3, this modification does not change the game, and therefore $\Pr[\mathsf{bad} \text{ in } G_5] = \Pr[\mathsf{bad} \text{ in } G_4]$.

Another syntactic change then leads us to G_6. Instead of re-sampling the keys $(K[1], \ldots, K[v], U[1, 0^\lambda], \ldots, U[v, 0^\lambda])$ at the end of the oracles E, E^{-1}, or RF, we shift this sampling to the beginning. As in Theorem 5.3, $\Pr[\mathsf{bad} \text{ in } G_6] = \Pr[\mathsf{bad} \text{ in } G_5]$.

In game G_6, we can now finally bound the probability of an adversary in provoking bad to be set, and we additionally observe that G_6 and G_7 differ only after bad is set. The probability $\Pr[\mathsf{bad} \text{ in } G_6]$ can be bounded as in Theorem 5.3 by

$$\Pr[\mathsf{bad} \text{ in } G_6] \leq \frac{(c+1)\gamma \cdot up(q_e \ell_{\text{blk}} + 1)}{(2^\lambda - p)(2^\kappa - p)} + \frac{2\gamma \cdot upq_e}{(2^\kappa - p)(2^\nu - pq_e)} + \frac{c\gamma \cdot p_i}{(2^\kappa - p)} + \frac{\gamma \cdot upq_e \cdot \ell_{\text{blk}}}{(2^\kappa - p)(2^\lambda - (q_e \ell_{\text{blk}} + 1))} + (p + uq_e \ell_{\text{blk}}) \left(\frac{1}{2^\lambda - uq_e \ell_{\text{blk}}}\right)^c \binom{uq_e \ell_{\text{blk}} + 1}{c+1},$$

so to conclude the proof we can now analyze $\Pr[G_7]$.

Then, following Lemma 6.1, (starting from G_1 and omitting the part on verification queries) we

obtain that this advantage is bounded by

$$\Pr[\mathrm{G}_7] \leq \frac{u(\ell_{\mathrm{blk}} q_e)^2 \cdot}{2^{\lambda+1}}.$$

∎

We stress that the term $up/2^{\kappa}$ in Equation (8) does not immediately correspond to a matching attack on the use of the scheme within the TLS protocol. The reason is that such an attack would require sending a great amount of crafted ciphertexts within the TLS session, but TLS tears down a session and discards the keys after the first failure in MAC verification. Therefore, it is conceivable that the scheme as used within TLS achieves considerably better security against active attacks than our above bound suggests. Moreover, such an attack would be inherently *active* and not suitable for mass surveillance.

6.4 Security of XCAU

To analyze the indistinguishability security of XCAU, we combine the results of Theorem 5.6 and Lemma 6.1. The proof is almost the same as the one for Theorem 6.2, but the step of "decoupling" the $\mathrm{E}/\mathrm{E}^{-1}$ and RF oracles makes use of the results in Theorem 5.6. Most notably and in contrast to RCAU, the bound does not contain a term of the type $p_i/2^{\kappa}$, and applies to active adversaries as well.

Theorem 6.6 *Let $\kappa, \lambda, \nu \geq 1$ be such that $\nu \leq \lambda - 2$. Let $\mathsf{H}\colon \{0,1\}^{\lambda} \times (\{0,1\}^* \times \{0,1\}^*) \to \{0,1\}^{\lambda}$ be an ϵ-almost XOR-universal hash function, for some $\epsilon\colon \mathbb{N} \times \mathbb{N} \to [0,1]$. Let $\mathsf{XCAU} = \mathbf{XCAU}[\mathsf{H}, \kappa, \lambda, \nu]$. Let A be an adversary that makes at most u queries to its* New *oracle, q_e queries per user instance to its* Enc *oracle with messages of length at most ℓ_{bit} bits, q_v queries per user instance to its* Vf *oracle with messages of length at most $\ell_{\mathrm{bit}} + \lambda$ bits, and p queries to its* E *and* E^{-1} *oracles. Assume furthermore that $q_e \leq 2^{\nu}$ and $\ell_{\mathrm{bit}} \leq \lambda(2^{\lambda-\nu} - 2)$. Then*

$$\mathsf{Adv}^{\text{mu-ind}}_{\mathsf{XCAU}}(A) \leq \frac{up(\ell_{\mathrm{blk}}(q_e+q_v)+1)}{2^{\lambda+\kappa+1}} + \frac{up(\ell_{\mathrm{blk}}(q_e+q_v)+1)^2}{2^{\lambda+1}} + uq_v\left(2^{-\lambda} + \epsilon(\ell_{\mathrm{bit}}, \ell_{\mathrm{head}})\right) + \frac{u(u-1)}{2^{\kappa+1}},$$

for $\ell_{\mathrm{blk}} = \lceil \ell_{\mathrm{bit}}/\lambda \rceil + 1$, and with headers of length at most $\lambda \ell_{\mathrm{head}}$ bits.

Proof: As in the previous proofs, we restrict ourselves to adversaries A that do not make trivially invalid queries. The first step, as in the proof of Theorem 5.7, is to change the game $\mathbf{G}^{\text{mu-ind}}_{\mathsf{XCAU}}(A)$ in an equivalent way by introducing an additional oracle RF that implements the same ideal cipher as E and E^{-1} and is used in the algorithms XCAU.Enc and XCAU.Dec, call this game G_0. The next step, still as in Theorem 5.7, is to re-write the oracles Enc, Dec, and RF such that the input and output whitening of the block cipher is performed within RF, this game is called G_1 and the changes are analogous to the modifications in G_1 in Theorem 5.7. The next game G_2 is then obtained by replacing RF with an independent instance of an ideal cipher, this is the same as G_1 in Theorem 6.2. Both games have a flag bad that is set when the oracles RF and $\mathrm{E}/\mathrm{E}^{-1}$ diverge.

The probability of setting the bad flag in G_1 and G_2 can be bounded using Theorem 5.6. More technically, we describe an adversary $B = B(A)$ that emulates oracles to A as follows: Queries New, E, and E^{-1} by B are responded by B performing the same query in its game. Queries Enc and Dec are responded by B emulating the respective oracles using the oracle RF in its game to evaluate CAU.Enc and CAU.Dec. The view of A is the same in G_1 and in the game $\mathrm{R}(B(A))$, and in G_2 and the game $\mathrm{S}(B(A))$, respectively. The numbers of queries u to the New oracle and p to the E and E^{-1} oracles are preserved by B. At most q_e queries of length at most ℓ_{bit} to Enc and

at most q_v queries of length at most $\ell_{\text{bit}} + \lambda$ to V_F translate into at most $\ell_{\text{blk}}(q_e + q_v) + 1$ queries to RF in the game played by B. Using the claim, this means that the probability of setting bad can be bounded by $up(\ell_{\text{blk}}(q_e + q_v) + 1)/2^{\lambda+\kappa+1}$.

We now conclude the proof similarly to the one in Theorem 6.2. We observe that game G_2 is almost equivalent to $\mathbf{G}^{\text{mu-ind}}_{\mathsf{CAU}}$ without access to the oracle E and E^{-1}, because these oracles are independent of all other oracles in the game, which makes them easy to simulate. We therefore describe an adversary $B = B(A)$ in the game $\mathbf{G}^{\text{mu-ind}}_{\mathsf{CAU}}$ that simulates the oracles E and E^{-1} to the adversary A and forwards all other queries to its own oracles. Clearly, $\Pr[\text{G}_2] = \Pr[\mathbf{G}^{\text{mu-ind}}_{\mathsf{CAU}}(B(A))]$, and therefore

$$\begin{aligned}
\mathsf{Adv}^{\text{mu-ind}}_{\mathsf{XCAU}}(A) &= 2\Pr[\mathbf{G}^{\text{mu-ind}}_{\mathsf{XCAU}}(A)] - 1 = 2\Pr[\text{G}_0] - 1 = 2\Pr[\text{G}_1] - 1 \\
&\leq 2\Pr[\mathbf{G}^{\text{mu-ind}}_{\mathsf{CAU}}(B(A))] - 1 + \frac{up(\ell_{\text{blk}}(q_e + q_v) + 1)}{2^{\lambda+\kappa+1}} \\
&\leq \frac{u(u-1)}{2^{\kappa+1}} + \frac{u((q_e + q_v) \cdot \ell_{\text{blk}})^2}{2^{\lambda+1}} + uq_v \left(2^{-\lambda} + \epsilon(\ell_{\text{bit}}, \ell_{\text{head}})\right) \\
&\quad + \frac{up(\ell_{\text{blk}}(q_e + q_v) + 1)}{2^{\lambda+\kappa+1}},
\end{aligned}$$

which concludes the proof. ∎

7 Conclusion

TLS 1.2 is the most widely used cryptographic protocol in the Internet, but due to issues with both performance and security, it will soon be replaced by its successor, TLS 1.3. Given that the bulk of Internet traffic will likely be protected by TLS 1.3 in the next years, it is extremely important that the security of the protocol is well-understood. Facing the threat of mass surveillance and the expected great number of TLS 1.3 sessions, the TLS Working Group has introduced a nonce-randomization technique to improve the resilience of TLS 1.3 against such attacks.

We show that the proposed technique can be understood as a key-length extension for AE; it essentially extends the 128-bit key of AES-GCM to a 224-bit key. We first describe the authenticated encryption CAU (Counter mode Almost Universal) as an abstraction of GCM. We then describe the scheme with randomized nonces as its variant RCAU and analyze it in the multi-user setting, where we show that it improves the resilience against (passive) mass surveillance as intended by the designers. We also show, however, that the AE does not perform as well as one might expect from an AE with a 224-bit key, especially in presence of active attacks. One alternative would be to simply increase the key size by, e.g., switching to an AES-256-based mode; this achieves better security but also impacts performance.

We suggest a new encryption mode that we call XCAU. The mode uses an additional 128-bit key (256 bits in total) to randomize the inputs and outputs of the block cipher (here AES) as in DESX. The mode is almost as efficient as the mode RCAU used in TLS 1.3, only adding two 128-bit xor operations for each call to the block cipher over plain CAU, our abstraction for GCM. We show that, still, its security is improved over RCAU in two ways. The security bounds we prove for security of XCAU are significantly better than those for RCAU, this stems mostly from the fact that *all* inputs to the block cipher are randomized. Furthermore, the whitening of the block-cipher output allows to remove the (for realistic parameters largest) term $p_i/2^\kappa$ from the security bound. (It should be noted, however, that this term is not worrisome for realistic parameters.) The fact that the implementation of XCAU, in contrast to that of RCAU, requires non-black-box changes

to the libraries implementing CAU, however, makes adoption in the currently developed standard TLS 1.3 difficult.

References

[1] C. Badertscher, C. Matt, U. Maurer, P. Rogaway, and B. Tackmann. Augmented secure channels and the goal of the TLS 1.3 record layer. In M. H. Au and A. Miyaji, editors, *ProvSec 2015*, volume 9451 of *LNCS*, pages 85–104. Springer, Heidelberg, Nov. 2015. 6

[2] M. Bellare, D. J. Bernstein, and S. Tessaro. Hash-function based PRFs: AMAC and its multi-user security. In *Advances in Cryptology–EUROCRYPT 2016*, pages 566–595. Springer, 2016. 6

[3] M. Bellare, A. Boldyreva, and S. Micali. Public-key encryption in a multi-user setting: Security proofs and improvements. In B. Preneel, editor, *EUROCRYPT 2000*, volume 1807 of *LNCS*, pages 259–274. Springer, Heidelberg, May 2000. 3, 6

[4] M. Bellare, R. Canetti, and H. Krawczyk. Pseudorandom functions revisited: The cascade construction and its concrete security. In *37th FOCS*, pages 514–523. IEEE Computer Society Press, Oct. 1996. 6

[5] M. Bellare and C. Namprempre. Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. In T. Okamoto, editor, *ASIACRYPT 2000*, volume 1976 of *LNCS*, pages 531–545. Springer, Heidelberg, Dec. 2000. 3, 8, 10

[6] M. Bellare and P. Rogaway. The security of triple encryption and a framework for code-based game-playing proofs. In Vaudenay [35], pages 409–426. 6, 12, 13, 30, 34

[7] M. Bellare and B. Tackmann. The multi-user security of authenticated encryption: AES-GCM in TLS 1.3. In M. Robshaw and J. Katz, editors, *Advances in Cryptology–CRYPTO 2016*, volume 9814 of *LNCS*, pages 247–276. Springer, 2016. 15

[8] D. J. Bernstein. Multi-user Schnorr security, revisited. Cryptology ePrint Archive, Report 2015/996, 2015. `http://eprint.iacr.org/2015/996`. 6

[9] M. K. Boyarsky. Public-key cryptography and password protocols: The multi-user case. In *ACM CCS 99*, pages 63–72. ACM Press, Nov. 1999. 6

[10] Y. Dodis, P. J. Lee, and D. H. Yum. Optimistic fair exchange in a multi-user setting. In T. Okamoto and X. Wang, editors, *PKC 2007*, volume 4450 of *LNCS*, pages 118–133. Springer, Heidelberg, Apr. 2007. 6

[11] M. Dworkin. Recommendation for block cipher modes of operation: The CCM mode for authentication and confidentiality. NIST Special Publication 800-38C, May 2004. 3

[12] M. Dworkin. Recommendation for block cipher modes of operation: Galois/Counter Mode (GCM) and GMAC. NIST Special Publication 800-38D, November 2007. 3, 6, 10

[13] S. Even and Y. Mansour. A construction of a cipher from a single pseudorandom permutation. *Journal of Cryptology*, 10(3):151–162, 1997. 5, 11

[14] M. Fischlin, F. Günther, G. A. Marson, and K. G. Paterson. Data is a stream: Security of stream-based channels. In R. Gennaro and M. J. B. Robshaw, editors, *CRYPTO 2015, Part II*, volume 9216 of *LNCS*, pages 545–564. Springer, Heidelberg, Aug. 2015. 6

[15] P.-A. Fouque, A. Joux, and C. Mavromati. Multi-user collisions: Applications to discrete logarithm, Even-Mansour and PRINCE. In P. Sarkar and T. Iwata, editors, *ASIACRYPT 2014, Part I*, volume 8873 of *LNCS*, pages 420–438. Springer, Heidelberg, Dec. 2014. 6

[16] S. Galbraith, J. Malone-Lee, and N. P. Smart. Public key signatures in the multi-user setting. *Information Processing Letters*, 83(5):263–266, 2002. 6

[17] Q. Huang, G. Yang, D. S. Wong, and W. Susilo. Efficient optimistic fair exchange secure in the multi-user setting and chosen-key model without random oracles. In T. Malkin, editor, *CT-RSA 2008*, volume 4964 of *LNCS*, pages 106–120. Springer, Heidelberg, Apr. 2008. 6

[18] T. Iwata, K. Ohashi, and K. Minematsu. Breaking and repairing GCM security proofs. In R. Safavi-Naini and R. Canetti, editors, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 31–49. Springer, Heidelberg, Aug. 2012. 6, 10, 29

[19] J. Kilian and P. Rogaway. How to protect DES against exhaustive key search (an analysis of DESX). *Journal of Cryptology*, 14(1):17–35, 2001. 5, 11

[20] E. Kiltz, D. Masny, and J. Pan. Optimal security proofs for signatures from identification schemes. Cryptology ePrint Archive, Report 2016/191, 2016. `http://eprint.iacr.org/`. 6

[21] H. Krawczyk. LFSR-based hashing and authentication. In Y. Desmedt, editor, *CRYPTO'94*, volume 839 of *LNCS*, pages 129–139. Springer, Heidelberg, Aug. 1994. 6

[22] T. Krovetz and P. Rogaway. The software performance of authenticated-encryption modes. In A. Joux, editor, *FSE 2011*, volume 6733 of *LNCS*, pages 306–327. Springer, Heidelberg, Feb. 2011. 3

[23] U. M. Maurer. Indistinguishability of random systems. In L. R. Knudsen, editor, *EUROCRYPT 2002*, volume 2332 of *LNCS*, pages 110–132. Springer, Heidelberg, Apr. / May 2002. 26

[24] D. A. McGrew and J. Viega. The security and performance of the Galois/counter mode (GCM) of operation. In A. Canteaut and K. Viswanathan, editors, *INDOCRYPT 2004*, volume 3348 of *LNCS*, pages 343–355. Springer, Heidelberg, Dec. 2004. 3, 6, 10, 29

[25] N. Mouha and A. Luykx. Multi-key security: The Even-Mansour construction revisited. In R. Gennaro and M. J. B. Robshaw, editors, *CRYPTO 2015, Part I*, volume 9215 of *LNCS*, pages 209–223. Springer, Heidelberg, Aug. 2015. 6, 11

[26] C. Namprempre, P. Rogaway, and T. Shrimpton. Reconsidering generic composition. In P. Q. Nguyen and E. Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 257–274. Springer, Heidelberg, May 2014. 3

[27] Y. Niwa, K. Ohashi, K. Minematsu, and T. Iwata. GCM security bounds reconsidered. In G. Leander, editor, *FSE 2015*, volume 9054 of *LNCS*, pages 385–407. Springer, Heidelberg, Mar. 2015. 6, 29

[28] P. Rogaway. Authenticated-encryption with associated-data. In V. Atluri, editor, *ACM CCS 02*, pages 98–107. ACM Press, Nov. 2002. 3, 7, 8

[29] P. Rogaway. Efficient instantiations of tweakable blockciphers and refinements to modes OCB and PMAC. In P. J. Lee, editor, *ASIACRYPT 2004*, volume 3329 of *LNCS*, pages 16–31. Springer, Heidelberg, Dec. 2004. 3

[30] P. Rogaway and M. Bellare. Robust computational secret sharing and a unified account of classical secret-sharing goals. In P. Ning, S. D. C. di Vimercati, and P. F. Syverson, editors, *ACM CCS 07*, pages 172–184. ACM Press, Oct. 2007. 6

[31] P. Rogaway, M. Bellare, J. Black, and T. Krovetz. OCB: A block-cipher mode of operation for efficient authenticated encryption. In *ACM CCS 01*, pages 196–205. ACM Press, Nov. 2001. 3

[32] P. Rogaway and T. Shrimpton. A provable-security treatment of the key-wrap problem. In Vaudenay [35], pages 373–390. 8

[33] B. Smith. Pull request: Removing the AEAD explicit IV. Mail to IETF TLS Working Group, March 2015. 4

[34] S. Tessaro. Optimally secure block ciphers from ideal primitives. In T. Iwata and J. H. Cheon, editors, *ASIACRYPT 2015, Part II*, volume 9453 of *LNCS*, pages 437–462. Springer, Heidelberg, Nov. / Dec. 2015. 6

[35] S. Vaudenay, editor. *EUROCRYPT 2006*, volume 4004 of *LNCS*. Springer, Heidelberg, May / June 2006. 42, 43