COBRA: A Parallelizable Authenticated Online Cipher without Block Cipher Inverse

Elena Andreeva[1,2], Atul Luykx[1,2], Bart Mennink[1,2], and Kan Yasuda[1,3]

[1] Department of Electrical Engineering, ESAT/COSIC, KU Leuven, Belgium.
[2] iMinds, Belgium.
[3] NTT Secure Platform Laboratories, Japan.

Abstract. We present a new, misuse-resistant scheme for online authenticated encryption, following the framework set forth by Fleischmann et al. (FSE 2012). Our scheme, COBRA, is roughly as efficient as the GCM mode of operation for nonce-based authenticated encryption, performing one block cipher call plus one finite field multiplication per message block in a parallelizable way. The major difference from GCM is that COBRA preserves privacy up to prefix under nonce repetition. However, COBRA only provides authenticity against nonce-respecting adversaries. As compared to COPA (ASIACRYPT 2013), our new scheme requires no block cipher inverse and hence enjoys provable security under a weaker assumption about the underlying block cipher. In addition, COBRA can possibly perform better than COPA on platforms where finite field multiplication can be implemented faster than the block cipher in use, since COBRA essentially replaces half of the block cipher calls in COPA with finite field multiplications.

Keywords. COPA, OTR, GCM, Feistel network, ManTiCore, authenticated online cipher, deterministic, finite-field multiplication.

1 Introduction

Authenticated encryption (AE) schemes target the security goals of privacy and integrity. The field of AE has received more interest in the light of the recently announced CAESAR competition [9]. In the target scope of the competition fall secure and efficient AE algorithms for specific or possibly multiple environments.

While AE can securely be achieved by combining a probabilistic encryption scheme and a message authentication code using Bellare and Namprempre's generic composition [6], this approach comes at the cost of using two keys, one for encryption and one for authentication. This and further efficiency optimization reasons have led to the development of many dedicated nonce-based AE solutions such as CCM [33], CWC [20], EAX [7], GCM [23], IACBC [18], IAPM [18], OCB1-3 [21, 27, 29], and OTR [24].

Of these schemes, today GCM is the most widely deployed. GCM has been standardized by many organizations including ANSI, IEEE, ISO/IEC, and NIST. GCM has also been adopted by major cryptographic protocols such as IPsec, SSH, and TLS/SSL.

One advantage of GCM is that it performs well on Intel CPUs. According to Gladman [13], GCM outperforms CCM, CWC, and EAX on Intel P3/P4 and AMD 64(32/64) processors, if a 64K table is used with GCM. This is mostly due to the fact that finite-field multiplication over GF(2^{128}) can be implemented efficiently on these platforms so that it runs faster than serial AES or hashing modulo $2^{127} - 1$.

There are several ways of parallelizing the polynomial hashing in GCM [14]. For example, instead of performing finite-field multiplications sequentially by Horner's rule as $\Big(\big((X[1]L \oplus X[2]\big)L \oplus X[3]\big)L \oplus X[4]\Big)L$, one precomputes L^2, L^3 and L^4, stores them in a table, and then computes the hash in a parallelizable way as $X[1]L^4 + X[2]L^3 + X[3]L^2 + X[4]L$. Here L denotes the key of polynomial hashing and $X[i]$ the data blocks.

On more recent Intel CPUs such as Nehalem and Sandy Bridge, finite-field multiplication runs slower than AES [21]. Note that these processors are equipped with dedicated instruction sets, PCLMULQDQ for finite-field multiplication and AES-NI for AES block cipher computation. However, according to the latest report by Gueron [15] PCLMULQDQ is now more efficient on the latest Haswell processor, making finite-field multiplication over GF(2^{128}) faster than AES block cipher computation. This also makes GCM still attractive for use on Intel platforms.

Another advantage of GCM is the fact that it does not require the block cipher inverse. This contrasts sharply with schemes like OCB, where the cipher inverse is necessary for decryption. Besides the extra cost to implement the inverse algorithm, the problem is that the security proof needs to rely on a stronger assumption about the underlying block cipher if its inverse is used by the scheme. This issue has been discussed for OCB [5] and has led to the invention of OTR [24].

Given these features and its wide-spread use, GCM is often considered as a reference AE mode of operation. In fact, the call for submissions of the CAESAR competition [9] requires that authors "must explain, in particular, why users should prefer this cipher over AES-GCM."

All of the above-mentioned dedicated schemes are proven secure in a nonce-respecting model — formalism proposed by Rogaway [28] — where an adversary is limited to making encryption queries only with non-repeating nonce values. For the cases when nonce values do repeat, none of these AE schemes provides any formal security guarantees. Indeed, all of these schemes, including the latest OTR, can be "attacked" under nonce repetition, as described by Fleischmann et al. [12].

Nonce repetition can, however, occur in practice due to the fact that the nonce is chosen by the application programmer rather than the scheme itself as discussed by Fleishmann et al. [12]. Examples of nonce repetition are flawed implementations [8, 10, 19, 22, 34], bad management of nonces by the user, and backup resets or virtual machine clones when the nonce is stored as a counter.

One way to address these situations is to design AE schemes which provide misuse resistance in a model where the adversary can perform queries with repeating nonces. Such schemes include the deterministic AE solutions SIV [30], BTM [16], and HBS [17], and also the authenticated online ciphers McOE-G [12], APE [2], and COPA [3]. The latter schemes are more efficient (need to process the message just once), even though the security under nonce repetition is limited to indistinguishability up to a common prefix.

We note that we are missing a "GCM-like" authenticated online cipher. McOE-G makes one block cipher call plus one finite-field multiplication per message block, but it is inherently sequential and not parallelizable like GCM. APE is permutation-based and sequential. COPA is parallelizable, but it makes two block cipher calls per message block. Moreover, all of these schemes require the inverse primitive calls for decryption. In this paper, therefore, we set out to propose a new authenticated online cipher whose efficiency is comparable to that of GCM.

Our Results. We present a secure and efficient solution for AE, which we name COBRA. A formal description of COBRA for integral message blocks is given in Sect. 3, and it is depicted in Figs. 2-3. (A description of COBRA for arbitrary-length messages is given in App. A.)

Design. At first glance our design may seem to combine characteristics of the COPA [3] and OTR [24] designs. Indeed, to ensure misuse-resistance we include features from COPA and then substitute the parallelization procedures with the two-round balanced Feistel structure as proposed by Minematsu [24] in OTR. The latter design decision enables the use of just a single type of primitive, namely a block cipher in the forward encryption direction, without losing parallelizability, for efficiently authenticating and encrypting at the same time using polynomial hashing. It also allows for a scheme that does not need the inverse of the block cipher in decryption.

However, the construction of COBRA is *not* motivated by the mere combination of the two designs. Indeed, the employment of the Feistel network seems *necessary* for efficiently authenticating and encrypting at the same time using polynomial hashing. It also allows for a scheme that does not need the inverse of the block cipher in decryption. In order to achieve integrity of COBRA, we utilize the checksum of intermediate state values of the Feistel structure, which is similar to a technique proposed by Anderson et al. in their ManTiCore design [1].

Security. In Sect. 4, we prove that COBRA is secure against chosen-plaintext attacks (CPA) and against forgery up to approximately $2^{n/2}$ queries, where n is the block length of the underlying cipher. Our result for privacy covers nonce-repeating attackers. This contrasts sharply with GCM whose security collapses once the nonce is repeated. Note that authenticity of COBRA requires nonce-respecting attackers.

Our new scheme requires no block cipher inverse and hence enjoys provable security under the pseudo-random permutation (PRP) assumption about the underlying block cipher. This is not the case for COPA, whose security proof relies on a stronger assumption about the block cipher.

Our proof itself is simplified by decomposing COBRA into smaller parts which are dealt with individually. The main idea here is to turn a call to the block cipher into a call to a tweakable cipher which we instantiate with Rogaway's XE [27] construction. COBRA utilizes universal hashing (finite-field multiplication) and produces the tag using intermediate values of the Feistel networks. These differences make COBRA's proof slightly simpler than that of COPA.

Efficiency. The efficiency of COBRA is comparable to that GCM. That is, they both perform one block cipher call plus one finite field multiplication per message block in a parallelizable way.

As compared to COPA, COBRA saves the cost of implementing the inverse of the underlying block cipher. COBRA performs potentially better than COPA on platforms where the finite-field multiplication runs faster than the underlying block cipher call. Such CPUs include Intel's latest Haswell processor, where a 128-bit multiplication using the PCLMULQDQ instruction set runs faster than one AES call even using the AES-NI instruction set, hence essentially faster than any other block cipher implemented.

Attack On Previous Scheme. In the period between acceptance and publication of this paper, Nandi found an attack on the authenticity of the scheme using a nonce-repeating adversary [25]. As a result we have reduced the security claim of authenticity from being secure against nonce-repeating adversaries to being secure against nonce-respecting adversaries and we have made a small adjustment in the processing of the nonce to accomplish this security level: instead of multiplying the nonce with the message blocks, we use it in the block cipher call to create the secret value L. This does not change the privacy proof and authenticity is achieved for the same reason that authenticity is achieved in OTR.

2 Preliminaries

By $(\{0,1\}^n)^+$ we denote the set of strings whose length is a positive multiple of n bits. Given two strings A and B, we use $A \parallel B$ and AB interchangeably to denote the concatenation of A and B. For $A \in \{0,1\}^*$, by $A10^*$ we denote the string with a 1 appended, and then padded with zeros until its length is a multiple of n. If X is a string with length a multiple of n, by $X[i]$ we denote the ith n-bit block of X. The length of a string X is denoted by $|X|$.

A block cipher $E : \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ is a function that takes as input a key $k \in \mathcal{K}$ and a plaintext $M \in \{0,1\}^n$, and produces a ciphertext $C = E(k, M)$. We sometimes write $E_k(\cdot) = E(k, \cdot)$. For a fixed key k, a block cipher is a permutation on n bits.

We can view the set $\{0,1\}^n$ of bit strings as the finite field $\mathrm{GF}(2^n)$ consisting of 2^n elements. To this end, we represent an element of $\mathrm{GF}(2^n)$ as a polynomial over the field $\mathrm{GF}(2)$ of degree less than n, and a string $a_{n-1}a_{n-2}\cdots a_1 a_0 \in \{0,1\}^n$ corresponds to the polynomial $a_{n-1}\mathtt{x}^{n-1} + a_{n-2}\mathtt{x}^{n-2} + \cdots + a_1\mathtt{x} + a_0 \in$

GF(2^n). The addition in the field is simply addition of polynomials over GF(2) (i.e., bitwise XOR, denoted by $\oplus$). To define multiplication in the field, we fix an irreducible polynomial $f(\mathtt{x})$ of degree n over the field GF(2). For $a(\mathtt{x}), b(\mathtt{x}) \in$ GF(2^n), their product is defined as $a(\mathtt{x})b(\mathtt{x}) \bmod f(\mathtt{x})$ — polynomial multiplication over the field GF(2) reduced modulo $f(\mathtt{x})$. We simply write $a(\mathtt{x})b(\mathtt{x})$ and $a(\mathtt{x}) \cdot b(\mathtt{x})$ to mean the product in the field GF(2^n), and denote the multiplication by $\otimes$.

The set $\{0,1\}^n$ can alternatively be regarded as a set of integers ranging from 0 through $2^n - 1$, where a string $a_{n-1}a_{n-2} \cdots a_1 a_0 \in \{0,1\}^n$ corresponds to the integer $a_{n-1}2^{n-1} + a_{n-2}2^{n-2} + \cdots + a_1 2 + a_0 \in [0, 2^n - 1]$. Based on these conversions, we often simply write elements of GF(2^n) as integers. For example, "2" means $\mathtt{x}$ and "3" means $\mathtt{x} + 1$. When we write multiplications such as $2 \cdot 3$, we mean those in the field GF(2^n).

3 Specification

In this section we give the specification of our scheme COBRA. Here we define COBRA for messages whose length is a positive multiple of $2n$, where n denotes the block length of the underlying block cipher. The case of fractional messages is given in App. A.

Let $E : \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ be an n-bit block cipher. COBRA consists of two functionalities, an encryption function $\mathcal{E}$ and a decryption function $\mathcal{D}$:

$$\mathcal{E} : \mathcal{K} \times \{0,1\}^{n-1} \times \{0,1\}^* \times (\{0,1\}^{2n})^+ \to (\{0,1\}^{2n})^+ \times \{0,1\}^n,$$
$$\mathcal{D} : \mathcal{K} \times \{0,1\}^{n-1} \times \{0,1\}^* \times (\{0,1\}^{2n})^+ \times \{0,1\}^n \to (\{0,1\}^{2n})^+ \cup \{\perp\}.$$

The function $\mathcal{E}$ takes as input a key K, a nonce N, associated data A, and a message M, and returns a ciphertext C and tag T: $(C, T) \leftarrow \mathcal{E}(K, N, A, M)$. The decryption function $\mathcal{D}$ also gets a ciphertext C and tag T in addition to a key, nonce and associated data; it outputs M if the tag is correct and $\perp$ otherwise, which we denote as $M/\perp \leftarrow \mathcal{D}(K, N, A, C, T)$.

On input of a key K, nonce N, associated data A, and a message M padded into n-bit blocks $M[1]M[2] \cdots M[2d]$ (resp., a ciphertext $C = C[1]C[2] \cdots C[2d]$ and tag T), the function $\mathcal{E}$ (resp., $\mathcal{D}$) is defined in Fig. 1. Note that the functions are sound: for any K, N, A, M we have $M \leftarrow \mathcal{D}(K, N, A, \mathcal{E}(K, N, A, M))$. For the case the associated data A is of length at most $4n$ and the message is of length at most $6n$, the function $\mathcal{E}$ is depicted in Figs. 2-3.

4 Security

We briefly settle some notation for the security analysis in Sect. 4.1. In Sect. 4.2, we introduce some preliminary results related to COBRA. Confidentiality of COBRA is then proven in Sect. 4.3, and integrity in Sect. 4.4.

```
COBRA-Encrypt $\mathcal{E}[E](N, A, M)$:
  $L \leftarrow E_k(N \| 1), \Sigma \leftarrow 0$
  $\tau \leftarrow 4L, V \leftarrow L$
  for $i = 1, \ldots, d$ do
    $V \leftarrow V \oplus M[2i-1]$
    $C[2i-1] \leftarrow V$
    $V \leftarrow (V \otimes L) \oplus M[2i]$
    $C[2i] \leftarrow V$
    $\rho \leftarrow E_k(\tau \oplus C[2i])$
    $\Sigma \leftarrow \Sigma \oplus \rho$
    $C[2i-1] \leftarrow \rho \oplus C[2i-1]$
    $\sigma \leftarrow E_k(\tau \oplus L \oplus C[2i-1])$
    $\Sigma \leftarrow \Sigma \oplus \sigma$
    $C[2i] \leftarrow \sigma \oplus C[2i]$
    if $i < d$ then
      $V \leftarrow V \otimes L$
      $\tau \leftarrow 2\tau$
    end if
  end for

  $U \leftarrow$ProcessAD$[E](A)$
  $T \leftarrow$ComputeTag$[E](L, \tau, \Sigma, N, U)$
  return $(C, T)$
```

```
COBRA-Decrypt $\mathcal{D}[E](N, A, C, T)$:
  $L \leftarrow E_k(N \| 1), \Sigma \leftarrow 0$
  $\tau \leftarrow 4L, V \leftarrow L$
  for $i = 1, \ldots, d$ do
    $\sigma \leftarrow E_k(\tau \oplus L \oplus C[2i-1])$
    $\Sigma \leftarrow \Sigma \oplus \sigma$
    $M[2i] \leftarrow \sigma \oplus C[2i]$
    $\rho \leftarrow E_k(\tau \oplus M[2i])$
    $\Sigma \leftarrow \Sigma \oplus \rho$
    $V' \leftarrow \rho \oplus C[2i-1]$
    $M[2i-1] \leftarrow V' \oplus V$
    $V' \leftarrow V' \otimes L$
    $V \leftarrow M[2i]$
    $M[2i] \leftarrow V' \oplus M[2i]$
    if $i < d$ then
      $V \leftarrow V \otimes L$
      $\tau \leftarrow 2\tau$
    end if
  end for

  $U \leftarrow$ProcessAD$[E](A)$
  $T' \leftarrow$ComputeTag$[E](L, \tau, \Sigma, N, U)$
  return $T = T'$ ? $M : \bot$
```

```
ProcessAD$[E](A)$:
  $X \leftarrow A \| 10^*$
  $J \leftarrow E_k(0)$
  $U \leftarrow J$
  for $i = 1, \ldots, |X|/n - 1$ do
    $U \leftarrow (U \oplus X[i]) \otimes J$
  end for
  $U \leftarrow E_k(2J \oplus U \oplus X[c])$
  return $U$
```

```
ComputeTag$[E](L, \tau, \Sigma, N, U)$:
  $\tau \leftarrow 3(\tau \oplus L)$
  $T \leftarrow E_k(\tau \oplus \Sigma)$
  $\tau \leftarrow 3\tau$
  $T \leftarrow E_k(\tau \oplus T \oplus N \oplus U)$
  return $T$
```

Fig. 1: COBRA.

4.1 Notation

When writing $x \xleftarrow{\$} X$ for some finite set X we mean that x is sampled uniformly from X. We write $\Pr[\mathsf{A} \mid \mathsf{B}]$ to denote the probability of event A given B.

Say that $M \in \{0,1\}^{2n\ell}$. We write $M[1]M[2]\cdots M[2\ell] \xleftarrow{n} M$ to denote the *blocks* that make up M, and $\hat{M}[1]\hat{M}[2]\cdots\hat{M}[\ell] \xleftarrow{2n} M$ to denote the *fragments* that make up M. Note that a fragment is made of two blocks: $\hat{M}[i] = M[2i-1] \| M[2i]$.

For convenience, we use the notation

$$\underset{\mathbf{D}}{\Delta}(f\,;\,g) := \left|\Pr[\mathbf{D}^f = 1] - \Pr[\mathbf{D}^g = 1]\right| \qquad (1)$$

to denote the distinguishing advantages of adversary $\mathbf{D}$ in distinguishing oracles f and g, where the notation $\mathbf{D}^{\mathcal{O}}$ indicates the value output by $\mathbf{D}$ after interacting with oracle $\mathcal{O}$. The probabilities are defined over the random coins used in the

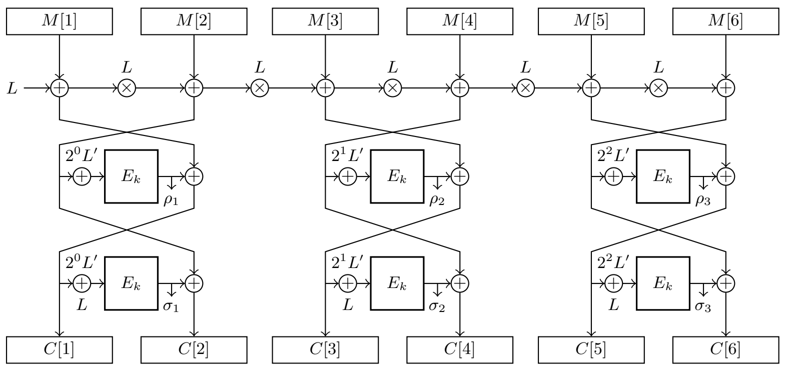

Fig. 2: Processing plaintext. Note that L' is defined in Fig. 3 below.

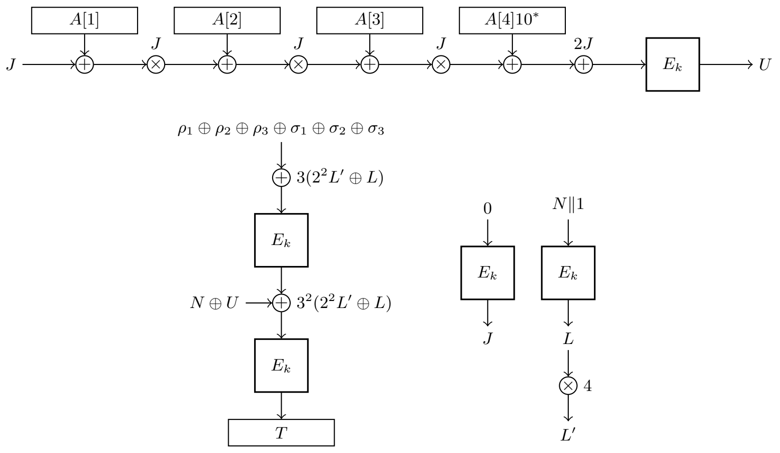

Fig. 3: Processing associated data (top), computing the tag (bottom left), and the secret values (bottom right).

oracles and the random coins of the adversary, if any. If a class of distinguishers is described by some parameters, e.g. the number of queries q, then by $\Delta_q(f\,;\,g)$ we denote the supremum of $\Delta_{\mathbf{D}}(f\,;\,g)$ over all distinguishers $\mathbf{D}$ in this class of adversaries. Multiple oracles are separated by a comma or given by a set, e.g. $\Delta(f_1, f_2\,;\,g_1, g_2)$ or $\Delta(\{f_1, f_2\}\,;\,\{g_1, g_2\})$ denotes distinguishing the combination of f_1 and f_2 from the combination of g_1 and g_2.

A uniform random function (URF) from m bits to n bits is a uniformly distributed random variable over the set of all functions from $\{0,1\}^m$ to $\{0,1\}^n$. A uniform random permutation (URP) on n bits is a uniformly distributed random variable over the set of all permutations on n bits.

Definition 1. *Let E be a block cipher. Let π be a URP on n bits. The* prp *advantage of a distinguisher* $\mathbf{D}$ *is defined as*

$$\mathbf{Adv}_E^{\mathrm{prp}}(\mathbf{D}) = \underset{\mathbf{D}}{\Delta}(E_k\,;\,\pi).$$

Here, $\mathbf{D}$ *is a distinguisher with oracle access to either* E_k *or* π*. The probabilities are taken over* $k \xleftarrow{\$} \mathcal{K}$*, the randomness of* π*, and random coins of* $\mathbf{D}$*, if any. By* $\mathbf{Adv}_E^{\mathrm{prp}}(t,q)$ *we denote the maximum advantage taken over all distinguishers that run in time* t *and make* q *queries.*

4.2 Preliminary Results

The input to each block cipher call in COBRA is first XORed with one of the following masks:

$$\{2J, 2^iL, 2^iL \oplus L, 3(2^iL \oplus L), 3^2(2^iL \oplus L)\}, \tag{2}$$

where $i \geq 2$, $J := E_k(0)$ and $L := E_k(1)$. As a result, each block cipher call can be viewed as a call to an XE construction [27]. Note that by using the doubling method from [27], we can produce many different values of the mask from the secret values J and L. Specifically, we adopt the tweaks used in [24], allowing us to replace each of the XEs with independent URFs.

Lemma 1 ([24, 27]). *Let* $\mathcal{T}$ *denote some set of indices such that* $\tau \to \mu_\tau$ *maps all indices to all tweaks injectively. The permutations* $\{E_k(\mu_\tau \oplus \cdot)\}_{\tau\in\mathcal{T}}$ *are indistinguishable from independent URFs* $\{\varphi_\tau\}_{\tau\in\mathcal{T}}$*. Specifically, let* $\mathbf{D}$ *be a distinguisher running in time* t *and making at most* q *queries, then*

$$\underset{\mathbf{D}}{\Delta}(\{E_k(\mu_\tau \oplus \cdot)\}_{\tau\in\mathcal{T}}\,;\,\{\varphi_\tau\}_{\tau\in\mathcal{T}}) \leq \frac{5q^2}{2^n} + \mathbf{Adv}_E^{\mathrm{prp}}(\mathbf{D}'),$$

where $\mathbf{D}'$ *is a distinguisher with running time similar to* $\mathbf{D}$*, making* $2q$ *queries.*

In Fig. 4 one can see a description of COBRA where the XE constructions are replaced with URFs, where the URFs are labeled α (replacing the XE construction in ProcessAD), β_i^N, γ_i^N for $i \geq 1$ and all N (replacing them in COBRA-Encrypt and COBRA-Decrypt), and δ_1^N, δ_2^N for all N (replacing them in ComputeTag). Throughout, we will denote this scheme by $\mathcal{E}'$.

We can describe $\mathcal{E}'$ as a sequence of functions each computing one ciphertext fragment, a function computing the tag, and a function processing the associated data. More formally:

Definition 2. *Say that* $\mathcal{E}'$ *maps* (N, A, M) *to* (C, T)*. Define* $f_i : \{0,1\}^n \times \{0,1\}^{2ni} \to \{0,1\}^{2n}$ *to be the function mapping* $(N, \hat{M}[1]\cdots\hat{M}[i])$ *to* $\hat{C}[i]$*,* $h : \{0,1\}^* \to \{0,1\}^n$ *the function mapping* A *to* U *(where* U *is as shown in Fig. 4), and* $f' : \{0,1\}^n \times \{0,1\}^* \times (\{0,1\}^{2n})^+ \to \{0,1\}^n$ *the function mapping* (N, A, M) *to* T*.*

As a second step, we replace the associated data computation h in $\mathcal{E}'$ with a URF Ω:

Lemma 2. *Let $\Omega : \{0,1\}^* \to \{0,1\}^n$ be a URF and let $\mathbf{D}$ be a distinguisher making at most q queries each of length less than nl, then*

$$\underset{\mathbf{D}}{\Delta}(h\,;\,\Omega) \le \frac{lq^2}{2^n}.$$

Proof. The URF α generates independent, uniformly distributed values as long as its inputs are unique. The only issue is when two different A's map to the same input to α, which itself reduces to finding zeros of a polynomial in J of degree at most l. Since J is an independent, uniformly distributed value generated using a URP, and polynomials of degree l have at most l distinct zeroes, the probability that a pair of plaintexts collides is $l/2^n$. By allowing the adversary to make q queries we get our desired bound. □

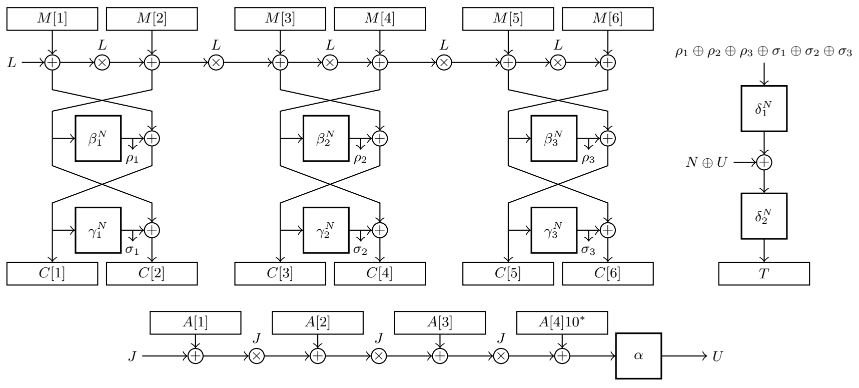

Fig. 4: Construction with independent URFs.

We define $\mathbb{E}$ to be $\mathcal{E}'$ with h replaced by Ω. In other words, $\mathbb{E}$ corresponds to COBRA where (i) the XE constructions have been replaced with independent URFs, and (ii) h has been replaced with Ω. Formally, we obtain the following result for $\mathbb{E}$.

Proposition 1. *Let $(\mathcal{E}, \mathcal{D})$ denote COBRA and let $\mathbf{D}$ be a distinguisher making at most q queries each of length less than $2n\ell$, then*

$$\underset{\mathbf{D}}{\Delta}(\mathcal{E}_k\,;\,\mathbb{E}) \le \frac{5(2\ell q + 2q)^2}{2^n} + \frac{2\ell q^2}{2^n} + \mathbf{Adv}_E^{\mathrm{prp}}(\mathbf{D}'),$$

where the probability is taken over $k \xleftarrow{\$} \mathcal{K}$ and the URFs in $\mathbb{E}$, and $\mathbf{D}'$ is a distinguisher with running time similar to $\mathbf{D}$, making $4\ell q + 4q$ queries.

Proof. We first apply Lem. 1, where we note that one query by $\mathbf{D}$ leads to at most $2\ell + 2$ block cipher calls, and then Lem. 2 to get the desired result. □

Using Prop. 1, we will prove confidentiality of COBRA in Sect. 4.3 and integrity in Sect. 4.4. For the proof of confidentiality, we present an additional elementary lemma:

Lemma 3. *Say f_1, g_1 are random functions independent of each other, and that f_2, g_2 are independent random functions as well. Let $\mathbf{D}$ be a distinguisher for $\{f_1, g_1\}$ and $\{f_2, g_2\}$ making q_f queries to the f_i oracles and q_g queries to the g_i oracle. Then there exist distinguishers $\mathbf{D}_f$ and $\mathbf{D}_g$ such that*

$$\underset{\mathbf{D}}{\Delta}(f_1, g_1\,;\, f_2, g_2) \le \underset{\mathbf{D}_f}{\Delta}(f_1\,;\, f_2) + \underset{\mathbf{D}_g}{\Delta}(g_1\,;\, g_2),$$

where $\mathbf{D}_f$ makes q_f queries and $\mathbf{D}_g$ makes q_g queries.

4.3 Confidentiality

We adopt the definitions of security given in [3], yet rather than comparing our scheme to a random variable over the set of all online permutations, we explicitly describe an ideal online function in terms of URFs.

Definition 3 (Ideal Online Function). *Let $g_i : \{0,1\}^n \times \{0,1\}^{2ni} \to \{0,1\}^{2n}$ be URFs and let $g' : \{0,1\}^n \times \{0,1\}^* \times (\{0,1\}^{2n})^+ \to \{0,1\}^n$ be a URF. We define $\$: \{0,1\}^n \times \{0,1\}^* \times (\{0,1\}^{2n})^+ \to (\{0,1\}^{2n})^+ \times \{0,1\}^n$ as*

$$\$(N, A, M) = g_1(N, \hat{M}[1]) \parallel g_2(N, \hat{M}[1]\hat{M}[2]) \parallel \cdots \parallel g_\ell(N, M) \parallel g'(N, A, M)$$

where $\hat{M}[1]\hat{M}[2] \cdots \hat{M}[\ell] \xleftarrow{2n} M$.

Definition 4 (IND-CPA). *Let $\mathcal{E}$ be an encryption scheme. The IND-CPA advantage of a distinguisher $\mathbf{D}$ relative to $\mathcal{E}$ is given by*

$$\mathbf{Adv}^{\mathrm{cpa}}_{\mathcal{E}}(\mathbf{D}) := \underset{\mathbf{D}}{\Delta}(\mathcal{E}_k\,;\, \$),$$

where $k \xleftarrow{\$} \mathcal{K}$ and $\$$ is as defined in Def. 3.

Theorem 1. *Let $\mathcal{E}$ denote COBRA and let $\mathbf{D}$ be a distinguisher running in time t and making at most q queries to $\mathcal{E}$, each of length less than $2n\ell$, then*

$$\mathbf{Adv}^{\mathrm{cpa}}_{\mathcal{E}}(\mathbf{D}) \le \frac{22(\ell+1)^2 q^2}{2^n} + \mathbf{Adv}^{\mathrm{prp}}_{E}(\mathbf{D}'),$$

where $\mathbf{D}'$ is a distinguisher with running time similar to $\mathbf{D}$, making $4\ell q + 4q$ queries.

Proof. Let $\mathbf{D}$ be a distinguisher running in time t and making at most q queries each of length less than $2n\ell$. As a first step, we move from $\mathcal{E}$ to $\mathbb{E}$, where the underlying XE constructions are replaced by independent URFs, and h by Ω. By Prop. 1:

$$\underset{\mathbf{D}}{\Delta}(\mathcal{E}_k\,;\,\mathbb{E}) \leq \frac{5(2\ell q+2q)^2}{2^n} + \frac{2q^2\ell}{2^n} + \mathbf{Adv}_E^{\mathrm{prp}}(\mathbf{D}'), \tag{3}$$

where $\mathbf{D}'$ has running time similar to $\mathbf{D}$ and makes at most $4\ell q + 4q$ queries. Next, note that the f_i's and f' (cf. Def. 2) are independent functions, as their underlying URFs $\beta_i^N, \gamma_i^N, \delta_i^N$ are independent functions. By Lem. 3:

$$\underset{\mathbf{D}}{\Delta}(\mathbb{E}\,;\,\$) \leq \underset{\mathbf{D}_t}{\Delta}(f'\,;\,g') + \sum_{i=1}^{\ell-1} \underset{\mathbf{D}_i}{\Delta}(f_i\,;\,g_i). \tag{4}$$

for some $\mathbf{D}_t$ that makes at most q queries of total length less than $2n\ell$ and $\mathbf{D}_i$ (for $i \in \{1,\ldots,\ell-1\}$) that makes at most q queries (of fixed length). A bound on $\Delta_{\mathbf{D}_i}(f_i\,;\,g_i)$ for arbitrary i is derived in Lem. 4. In Lem. 5 we compute a bound on $\Delta_{\mathbf{D}_t}(f'\,;\,g')$. We find:

$$\underset{\mathbf{D}}{\Delta}(\mathbb{E}\,;\,\$) \leq \frac{q^2(2\ell+3)}{2^n} + \sum_{i=1}^{\ell-1} \frac{q^2 2(2i+1)}{2^n} = \frac{q^2(2\ell+3)}{2^n} + \frac{q^2 2(\ell^2-1)}{2^n}. \tag{5}$$

The proof is completed by simplifying the obtained bound. □

Lemma 4. *Let $\mathbf{D}_i$ be a distinguisher making at most q queries, then*

$$\underset{\mathbf{D}_i}{\Delta}(f_i\,;\,g_i) \leq \frac{2(2i+1)q^2}{2^n}.$$

Proof. The proof is similar to that of Lem. 2. We use that the inputs to the URFs are polynomials of degree at most $2i+1$, and that f_i consists of two URFs β_i^N and γ_i^N. □

Lemma 5. *Let $\mathbf{D}_t$ be a distinguisher making at most q queries each of length less than $2n\ell$, then*

$$\underset{\mathbf{D}_t}{\Delta}(f'\,;\,g') \leq \frac{(2\ell+3)q^2}{2^n}.$$

Proof. Without loss of generality, we may assume that the distinguisher does not make repeat queries. Say that $\{(N_1,A_1,M_1),\ldots,(N_i,A_i,M_i)\}$ is the query history. We consider what happens on query (N^*,A^*,M^*).

Let $U_i := \Omega(A_i)$ and let Σ_i denote input to $\delta_1^{N_i}$ (see Fig. 4). Let U^* and Σ^* be the corresponding values for (N^*,A^*,M^*). We compute the probability that

$$U^* \oplus \delta_1^{N^*}(\Sigma^*) \oplus N^* = U_j \oplus \delta_1^{N_j}(\Sigma_j) \oplus N_j \tag{6}$$

for some j for which $1 \leq j \leq i$.

1. If $A^* \neq A_j$ then U^* is independent of $U_j \oplus \delta_1^{N_j}(\Sigma_j) \oplus \delta_1^{N^*}(\Sigma^*) \oplus N^* \oplus N_j$, hence the probability that equation (6) is satisfied is not more than $1/2^n$.
2. If $A^* = A_j$, then $U^* = U_j$ and we focus on the probability that

$$\delta_1^{N^*}(\Sigma^*) \oplus N^* = \delta_1^{N_j}(\Sigma_j) \oplus N_j. \tag{7}$$

   (a) If $M^* = M_j$, then $\Sigma^* = \Sigma_j$ and equation (7) reduces to $\delta_1^{N^*}(\Sigma^*) \oplus \delta_1^{N_j}(\Sigma^*) = N^* \oplus N_j$. Since we do not allow repeat queries $N^* \neq N_j$, and so this occurs with probability $1/2^n$.
   (b) Say that $M^* \neq M_j$, and let ρ^* and ρ_j denote the output of the last call to β made when processing M^* and M_j, respectively. If ρ^* and ρ_j are independent, then $\Sigma^* = \Sigma_j$ with probability $1/2^n$. If $\Sigma^* \neq \Sigma_j$, then $\delta_1^{N^*}(\Sigma^*)$ and $\delta_1^{N_j}(\Sigma_j)$ are independent (and also independent of N^* and N_j), hence the probability of equation (7) being true is upper bounded by $2/2^n$.
   The probability that ρ^* and ρ_j are not independent is upper bounded by the probability of having a collision in the inputs to the last β call (and only if M^* and M_j are the same length), which is $(2\ell + 1)/2^n$.

Putting our results together we get that

$$\Pr(\text{equation (6) holds}) \leq \max\left\{\frac{1}{2^n}, \frac{2\ell+1}{2^n} + \frac{2}{2^n}\right\} = \frac{2\ell+3}{2^n}. \tag{8}$$

This means that the probability that $U^* \oplus \delta_1^{N^*}(\Sigma^*) \oplus N^*$ collides with any of the previous $U_j \oplus \delta_1^{N_j}(\Sigma_j) \oplus N_j$ is upper bounded by $\frac{q(2\ell+3)}{2^n}$. As long as the input to δ_2 is unique, the tag produced is uniform and independent of all previous values, hence f' remains indistinguishable from g'. Summing over all queries, we get the desired bound. □

4.4 Integrity

Definition 5. *Let $\mathcal{E}$ be an AE scheme. The integrity advantage of a distinguisher $\mathbf{D}$ relative to $\mathcal{E}$ is given by*

$$\mathbf{Adv}_{\mathcal{E}}^{\text{int}}(\mathbf{D}) := \underset{\mathbf{D}}{\Delta}(\mathcal{E}_k, \mathcal{D}_k\,;\, \mathcal{E}_k, \perp),$$

where $k \xleftarrow{\$} \mathcal{K}$ and $\perp$ is a function that responds with $\perp$ on every query. We assume that the distinguisher does not make queries of the form $\mathcal{D}_k(N, A, C, T)$, where $(C, T) = \mathcal{E}_k(N, A, M)$ for some previously queried (N, A, M) and that it does not query $\mathcal{E}_k$ twice under the same nonce.

Theorem 2. *Let $\mathcal{E}$ denote COBRA and let $\mathbf{D}$ be a distinguisher running in time t and making at most q queries to $\mathcal{E}$ and q_f forgery attempts, each of length less than $2n\ell$, then*

$$\mathbf{Adv}_{\mathcal{E}}^{\text{int}}(\mathbf{D}) \leq \frac{(3q+1)q_f}{2^n} + \frac{22(\ell+1)^2 q^2}{2^n} + \mathbf{Adv}_E^{\text{prp}}(\mathbf{D}'),$$

where $\mathbf{D}'$ *is a distinguisher with running time similar to* $\mathbf{D}$*, making* $4\ell q + 4q$ *queries.*

Proof. As with the proof of confidentiality, we use Prop. 1 to switch to $\mathbb{E}$.

We first focus on adversaries with one forgery attempt, with (N^*, A^*, C^*, T^*) being the attempt. Let $\{(N_1, A_1, M_1), \dots, (N_q, A_q, M_q)\}$ denote the history of queries made by the adversary to the encryption oracle, where (C_i, T_i) is the output corresponding to (N_i, A_i, M_i) and each N_i is distinct. Let $U_i := \Omega(A_i)$, $U^* := \Omega(A^*)$, and let Σ_i denote the input to $\delta_1^{N_i}$ during the computation of (N_i, A_i, M_i); define Σ^* similarly. Note that

$$\delta_2^{N_i}\Big(U_i \oplus \delta_1^{N_i}(\Sigma_i) \oplus N_i\Big) = T_i, \tag{9}$$

and similarly for T^*.

If

$$U^* \oplus \delta_1^{N^*}(\Sigma^*) \oplus N^* \neq U_i \oplus \delta_1^{N_i}(\Sigma_i) \oplus N_i \tag{10}$$

for all i, then the input to δ_2 is distinct from all previous inputs to δ_2, hence the output of δ_2 from the forgery query is uniformly distributed and independent of T^*, which means that the forgery will be successful with probability at most $1/2^n$. Hence we focus on computing the probability that there is an i resulting in a collision in the δ_2 input.

Fix an i such that $1 \leq i \leq q$. We compute the probability that

$$U^* \oplus \delta_1^{N^*}(\Sigma^*) \oplus N^* = U_i \oplus \delta_1^{N_i}(\Sigma_i) \oplus N_i. \tag{11}$$

1. If $A^* \neq A_i$, then U^* is uniformly distributed and independent of U_i, hence the probability that equation (11) is satisfied is bounded above by $1/2^n$.
2. If $A^* = A_i$, then $U^* = U_i$ and we focus on the probability that

$$\delta_1^{N^*}(\Sigma^*) \oplus N^* = \delta_1^{N_i}(\Sigma_i) \oplus N_i. \tag{12}$$

   (a) If $C^* = C_j$, equation (12) reduces to

$$\delta_1^{N^*}(\Sigma^*) \oplus N^* = \delta_1^{N_i}(\Sigma^*) \oplus N_i. \tag{13}$$

   Since $A^* = A_j$, then either $N^* \neq N_i$, in which case we only get a successful forgery with probability $1/2^n$, or $N^* = N_j$ and $T^* \neq T_j$, in which case we get a failed forgery attempt as well.

   (b) Say that $C^* \neq C_j$, and that they differ at the mth fragment, i.e. $\hat{C}^*[m] \neq \hat{C}_j[m]$. We also assume that $N^* = N_j$, because if $N^* \neq N_j$ then the tags are independent of each other since they are produced by independent URFs $\delta_2^{N^*}$ and $\delta_2^{N_j}$.
   Since $N^* = N_j$, and N_j does not equal any of the other N_i, Σ^* is independent of all Σ_i for $i \neq j$. If $C^*[2m-1] = C_j[2m-1]$, then $C^*[2m] \neq C_j[2m]$, hence the inputs to $\beta_m^{N^*}$ for C^* and C_j are different. This means that $\Sigma^* = \Sigma_j$ with probability at most $1/2^n$, hence $\delta_1(\Sigma^*) = \delta_1(\Sigma_j)$ with

probability at most $2/2^n$. If $\delta_1^{N^*}(\Sigma^*) \neq \delta_1^{N_j}(\Sigma_j)$, then equation (12) is satisfied with probability at most $1/2^n$ since N^* and N_i are independent of the outputs of δ_1. If $C^*[2m-1] \neq C_j[2m-1]$, we can apply the same reasoning.

Putting the above results together, we get that the probability of equation (11) being satisfied is bounded above by $3/2^n$. Hence, the probability that there exists an i satisfying (11) is bounded above by $3q/2^n$. The probability that the forgery is successful is thus bounded above by $3q/2^n + 1/2^n$.

Generalizing to adversaries which can make up to q_f forgery queries as explained in Andreeva et al. [4], we have our desired bound. □

5 Future Work

We shall implement COBRA and compare its software performance with GCM and COPA. It is interesting to see how much of an overhead COBRA actually has over GCM on a specific platform, possibly due to the Feistel network, larger state, extra mask generation, and the reverse order of multplication and block cipher call.

Acknowledgments. The authors would like to thank FSE 2014 reviewers for their valuable comments. This work has been funded in part by the IAP Program P6/26 BCRYPT of the Belgian State (Belgian Science Policy), in part by the European Commission through the ICT program under contract ICT-2007-216676 ECRYPT II, in part by the Research Council KU Leuven: GOA TENSE, and in part by the Research Fund KU Leuven, OT/08/027. Elena Andreeva is supported by a Postdoctoral Fellowship from the Flemish Research Foundation (FWO-Vlaanderen). Bart Mennink and Atul Luykx are supported by Ph.D. Fellowships from the Institute for the Promotion of Innovation through Science and Technology in Flanders (IWT-Vlaanderen).

References

1. Anderson, E., Beaver, C.L., Draelos, T., Schroeppel, R., Torgerson, M.: ManTiCore: Encryption with Joint Cipher-State Authentication. In: Wang, H., Pieprzyk, J., Varadharajan, V. (eds.) ACISP. Lecture Notes in Computer Science, vol. 3108, pp. 440–453. Springer (2004)
2. Andreeva, E., Bogdanov, A., Luykx, A., Mennink, B., Mouha, N., Yasuda, K.: APE: Authenticated Permutation-Based Encryption for Lightweight Cryptography. In: FSE. Lecture Notes in Computer Science, Springer (2014), to appear
3. Andreeva, E., Bogdanov, A., Luykx, A., Mennink, B., Tischhauser, E., Yasuda, K.: Parallelizable and authenticated online ciphers. In: Sako, K., Sarkar, P. (eds.) ASIACRYPT (1). Lecture Notes in Computer Science, vol. 8269, pp. 424–443. Springer (2013)

4. Andreeva, E., Bogdanov, A., Luykx, A., Mennink, B., Tischhauser, E., Yasuda, K.: Parallelizable and authenticated online ciphers. Cryptology ePrint Archive (2013), full version of this paper
5. Aoki, K., Yasuda, K.: The security of the OCB mode of operation without the SPRP assumption. In: Susilo, W., Reyhanitabar, R. (eds.) ProvSec 2013. Lecture Notes in Computer Science, vol. 8209, pp. 202–220. Springer (2013)
6. Bellare, M., Namprempre, C.: Authenticated Encryption: Relations among Notions and Analysis of the Generic Composition Paradigm. In: Okamoto, T. (ed.) ASIACRYPT. Lecture Notes in Computer Science, vol. 1976, pp. 531–545. Springer (2000)
7. Bellare, M., Rogaway, P., Wagner, D.: The EAX Mode of Operation. In: Roy and Meier [32], pp. 389–407
8. Borisov, N., Goldberg, I., Wagner, D.: Intercepting mobile communications: the insecurity of 802.11. In: Rose, C. (ed.) MOBICOM. pp. 180–189. ACM (2001)
9. CAESAR: Competition for Authenticated Encryption: Security, Applicability, and Robustness (April 2013), `http://competitions.cr.yp.to/caesar.html`
10. Cantero, H.M., Peter, S., Bushing, Segher: Console Hacking 2010 – PS3 Epic Fail. 27th Chaos Communication Congress (December 2010)
11. Chakraborty, D., Sarkar, P.: Hch: A new tweakable enciphering scheme using the hash-counter-hash approach. IEEE Transactions on Information Theory 54(4), 1683–1699 (2008)
12. Fleischmann, E., Forler, C., Lucks, S.: McOE: A Family of Almost Foolproof On-Line Authenticated Encryption Schemes. In: Canteaut, A. (ed.) FSE. Lecture Notes in Computer Science, vol. 7549, pp. 196–215. Springer (2012)
13. Gladman, B.: AES and combined encryption/authentication modes. `http://www.gladman.me.uk/` (2006)
14. Gopal, V., Ozturk, E., Feghali, W., Guilford, J., Wolrich, G., Dixon, M.: Optimized Galois-Counter-Mode implementation on Intel® architecture processors. Intel Corporation White Paper (2010)
15. Gueron, S.: AES-GCM software performance on the current high end CPUs as a performance baseline for CAESAR competition. Directions in Authenticated Ciphers (DIAC) (2013)
16. Iwata, T., Yasuda, K.: BTM: A Single-Key, Inverse-Cipher-Free Mode for Deterministic Authenticated Encryption. In: Jacobson Jr, M.J., Rijmen, V., Safavi-Naini, R. (eds.) Selected Areas in Cryptography. Lecture Notes in Computer Science, vol. 5867, pp. 313–330. Springer (2009)
17. Iwata, T., Yasuda, K.: HBS: A Single-Key Mode of Operation for Deterministic Authenticated Encryption. In: Dunkelman, O. (ed.) FSE. Lecture Notes in Computer Science, vol. 5665, pp. 394–415. Springer (2009)
18. Jutla, C.S.: Encryption Modes with Almost Free Message Integrity. J. Cryptology 21(4), 547–578 (2008)
19. Kohno, T.: Attacking and Repairing the WinZip Encryption Scheme. In: Atluri, V., Pfitzmann, B., McDaniel, P.D. (eds.) ACM Conference on Computer and Communications Security. pp. 72–81. ACM (2004)
20. Kohno, T., Viega, J., Whiting, D.: CWC: A High-Performance Conventional Authenticated Encryption Mode. In: Roy and Meier [32], pp. 408–426
21. Krovetz, T., Rogaway, P.: The Software Performance of Authenticated-Encryption Modes. In: Joux, A. (ed.) FSE. Lecture Notes in Computer Science, vol. 6733, pp. 306–327. Springer (2011)

22. Lenstra, A.K., Hughes, J.P., Augier, M., Bos, J.W., Kleinjung, T., Wachter, C.: Public Keys. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO. Lecture Notes in Computer Science, vol. 7417, pp. 626–642. Springer (2012)
23. McGrew, D.A., Viega, J.: The Security and Performance of the Galois/Counter Mode (GCM) of Operation. In: Canteaut, A., Viswanathan, K. (eds.) INDOCRYPT. Lecture Notes in Computer Science, vol. 3348, pp. 343–355. Springer (2004)
24. Minematsu, K.: Parallelizable Rate-1 Authenticated Encryption from Pseudorandom Functions. In: EUROCRYPT. Lecture Notes in Computer Science, Springer (2014), to appear
25. Nandi, M.: Forging Attack on COBRA. Cryptographic Competitions Google Group (2014), `https://groups.google.com/d/msg/crypto-competitions/nhqcgEThcPc/ryvKY7lfMhMJ`
26. Ristenpart, T., Rogaway, P.: How to Enrich the Message Space of a Cipher. In: Biryukov, A. (ed.) FSE. Lecture Notes in Computer Science, vol. 4593, pp. 101–118. Springer (2007)
27. Rogaway, P.: Efficient Instantiations of Tweakable Blockciphers and Refinements to Modes OCB and PMAC. In: Lee, P.J. (ed.) ASIACRYPT. Lecture Notes in Computer Science, vol. 3329, pp. 16–31. Springer (2004)
28. Rogaway, P.: Nonce-Based Symmetric Encryption. In: Roy, B.K., Meier, W. (eds.) FSE 2004. Lecture Notes in Computer Science, vol. 3017, pp. 348–359. Springer (2004)
29. Rogaway, P., Bellare, M., Black, J., Krovetz, T.: OCB: a block-cipher mode of operation for efficient authenticated encryption. In: Reiter, M.K., Samarati, P. (eds.) ACM Conference on Computer and Communications Security. pp. 196–205. ACM (2001)
30. Rogaway, P., Shrimpton, T.: A Provable-Security Treatment of the Key-Wrap Problem. In: Vaudenay, S. (ed.) EUROCRYPT 2006. Lecture Notes in Computer Science, vol. 4004, pp. 373–390. Springer (2006)
31. Rogaway, P., Wooding, M., Zhang, H.: The Security of Ciphertext Stealing. In: Canteaut, A. (ed.) FSE 2012. Lecture Notes in Computer Science, vol. 7549, pp. 180–195. Springer (2012)
32. Roy, B.K., Meier, W. (eds.): Fast Software Encryption, 11th International Workshop, FSE 2004, Delhi, India, February 5-7, 2004, Revised Papers, Lecture Notes in Computer Science, vol. 3017. Springer (2004)
33. Whiting, D., Housley, R., Ferguson, N.: AES Encryption and Authentication Using CTR Mode and CBC-MAC. IEEE 802.11-02/001r2 (2002)
34. Wu, H.: The Misuse of RC4 in Microsoft Word and Excel. Cryptology ePrint Archive, Report 2005/007 (2005)

A COBRA for Arbitrary-Length Messages

We use ciphertext stealing [31] in order to deal with messages of arbitrary length. Let M be a message where $M[1]M[2]\cdots M[2\ell-1]M[2\ell] = M$ and $|M[i]| = n$ for $1 \le i < 2\ell - 1$.

A.1 $\ell > 1$, $|M[2\ell-1]| = n$, and $0 < |M[2\ell]| < n$

We start by computing the ciphertext of $M[1]\cdots M[2\ell-2]$ as is usually done in COBRA, resulting in $C[1]\cdots C[2\ell-2]$. Let M^* denote the rightmost $|M[2\ell]|$

bits of $C[2\ell-2]$, and we write $C[2\ell-2] = C'[2\ell-2]M^*$. Then we compute the final ciphertext fragment $C[2\ell-1]C[2\ell]$ using $M[2\ell-1]M[2\ell]M^*$ as our "new" final message fragment, using different tweaks for the final block cipher calls. The resulting ciphertext is

$$C[1]\cdots C[2\ell-3]C'[2\ell-2]C[2\ell-1]C[2\ell]. \tag{14}$$

Fig. 5 shows a diagram of the process. Note that we can recover M^* with just knowledge of $C[2\ell-1]$ and $C[2\ell]$:

$$\begin{aligned} M[2\ell]M^* = &\Big[C[2\ell] \oplus E_{k,\tau_2}(C[2\ell-1])\Big] \oplus \\ &\qquad \Big(\Big[E_{k,\tau_1}\big(C[2\ell] \oplus E_{k,\tau_2}(C[2\ell-1])\big) \oplus C[2\ell-1]\Big] \otimes L\Big), \end{aligned}$$

where $E_{k,\tau_1}(x) := E_k(x \oplus 7 \cdot 2^\ell L')$ and $E_{k,\tau_2}(x) := E_k(x \oplus 7 \cdot (2^\ell L' \oplus L))$.

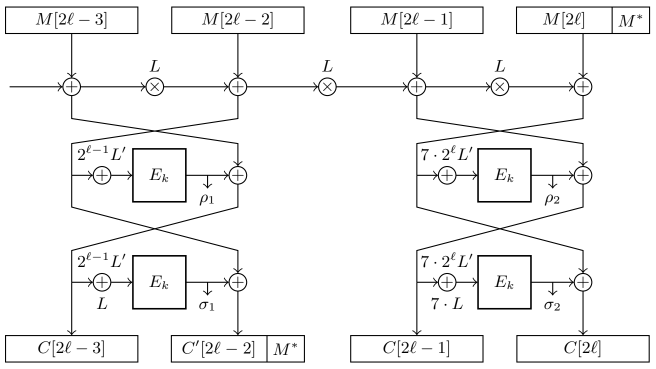

Fig. 5: Messages where the last block is not of full length, i.e. $0 < |M[2\ell]| < n$. Here M^* is "stolen" from ciphertext block $C[2\ell-2]$ and used in the input to the final fragment.

A.2 $\ell > 2$ and $0 < |M[2\ell-1]| \le n$

When there is no last block $M[2\ell]$, we replace it with the preceding ciphertext block, $C[2\ell-2]$. Then we steal ciphertext M^* of length $|M[2\ell-1]|$ from the ciphertext block $C[2\ell-4]$ such that $C[2\ell-4] = C'[2\ell-4]M^*$. The rest of the computation is similar to the previous case (Sect. A.1) and is depicted in Fig. 6.

A.3 $|M| \le 3n$

The above methods only work for messages of length greater than $3n$ (otherwise there is no ciphertext to steal from). We need to use different techniques in order to deal with shortest messages.

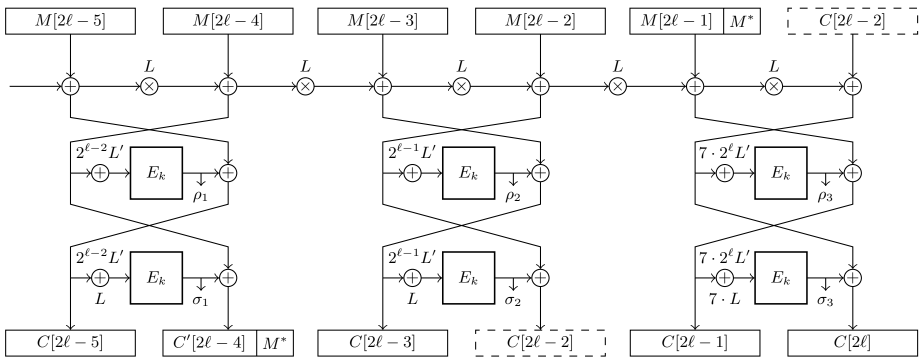

Fig. 6: Messages where the last fragment is of length less than or equal to n, i.e. $0 < |M[2\ell-1]| \leq n$. Here M^* is stolen from ciphertext block $C[2\ell-4]$ and used in the input to the final fragment together with ciphertext fragment $C[2\ell-2]$.

For $2n < |M| \leq 3n$ we can use a technique similar as to what is used in COPA [3]. Instead of using XLS [26] which uses the inverse block cipher, we can use HCH [11] in order to compute the output as follows:

$$C[1]C[2]T' \leftarrow \mathcal{E}(M[1]M[2]) \tag{15}$$

$$C[3]T \leftarrow \mathrm{HCH}(M[3]T'), \tag{16}$$

where $\mathcal{E}$ denotes COBRA and the final output of the scheme is $C[1]C[2]C[3]T$.

For $n < |M| < 2n$ we can use the tag-splitting method: we first compute

$$C[1]C[2]T \leftarrow \mathcal{E}(M[1]M[2]10^*), \tag{17}$$

then remove part of the tag so that the length of the output is equal to the length of the input. Here, again, $\mathcal{E}$ is COBRA, except different tweaks must be used from the case in which $|M| = 2n$.