Submission to the CAESAR competition

AES-COPA v.1

Designers/Submitters:
Elena Andreeva[1,2], Andrey Bogdanov[3], Atul Luykx[1,2],
Bart Mennink[1,2], Elmar Tischhauser[3], and Kan Yasuda[1,4]

Affiliation:
[1] Dept. Electrical Engineering, ESAT/COSIC, KU Leuven, Belgium.
[2] iMinds, Belgium.
[3] DTU Compute, Technical University of Denmark, Denmark.
[4] NTT Secure Platform Laboratories, Japan.

copa@esat.kuleuven.be

1 Specification

1.1 Parameters

AES-COPA has two parameters: the key length κ and the tag length τ. The key length can be either 16 bytes (128 bits), 24 bytes (192 bits), or 32 bytes (256 bits). The tag length is between 8 bytes (64 bits) and 16 bytes (128 bits). The nonce, also called the public message number, is an input of length 16 bytes (128 bits).

AES-COPA does not support a secret message number. Each key size of AES-COBRA corresponds to a key size of AES. AES-COPA supports variable length associated data and plaintexts. To comply with our security claims from Sect. 2, the length of the associated data together with the plaintext data is at most $\approx 2^{64} \cdot 16$ bytes.

Recommended parameter set: 16 byte (128 bits) key length and 16 byte (128 bits) tag length.

1.2 Notation

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. Throughout the document E denotes the block cipher AES-128 and n denotes its block size (128 bits). Strings of length n are called blocks and strings of length $2n$ are called fragments.

By $\{0,1\}^*$ we denote the set of all strings, and by $\{0,1\}^+$ the set of all non-empty strings. 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 $|X|$. By $\lfloor X \rfloor_j$ we denote the j most significant bits of X.

We can view the set $\{0,1\}^n$ of bit strings as the finite field GF(2^n) consisting of 2^n elements. To this end, we represent an element of GF(2^n) as a polynomial over the field GF(2) of degree less than n, and a string $a_{n-1}a_{n-2}\cdots a_1a_0 \in \{0,1\}^n$ corresponds to the polynomial $a_{n-1}\mathtt{x}^{n-1} + a_{n-2}\mathtt{x}^{n-2} + \cdots + a_1\mathtt{x} + a_0 \in \mathrm{GF}(2^n)$. Addition in the field is addition of polynomials over GF(2) (i.e. bitwise XOR, denoted by $\oplus$). To define multiplication in the field, we fix an irreducible polynomial $f(\mathtt{x}) := \mathtt{x}^{128} + \mathtt{x}^7 + \mathtt{x}^2 + \mathtt{x} + 1$ over the field GF(2). For $a(\mathtt{x}), b(\mathtt{x}) \in \mathrm{GF}(2^n)$, 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 be also regarded as a set of integers ranging from 0 through $2^n - 1$. A string $a_{n-1}a_{n-2}\cdots a_1a_0 \in \{0,1\}^n$ corresponds to the integer $a_{n-1}2^{n-1} + a_{n-2}2^{n-2} + \cdots + a_1 2 + a_0 \in [0, 2^n - 1]$. We often write elements of GF(2^n) as integers, based on these conversions. So, for example, "2" means $\mathtt{x}$, "3" means $\mathtt{x} + 1$, and "7" means $\mathtt{x}^2 + \mathtt{x} + 1$. When we write multiplications such as $2 \cdot 3$ and 7^2, we mean those in the field GF(2^n).

1.3 Authenticated Encryption

The encryption $\mathcal{E}$ and decryption $\mathcal{D}$ functions of AES-COPA have the following interfaces:

$$\mathcal{E} : \{0,1\}^\kappa \times \{0,1\}^n \times \{0,1\}^* \times \{0,1\}^+ \to \{0,1\}^+ \times \{0,1\}^\tau,$$
$$\mathcal{D} : \{0,1\}^\kappa \times \{0,1\}^n \times \{0,1\}^* \times \{0,1\}^+ \times \{0,1\}^\tau \to \{0,1\}^+ \cup \{\perp\}.$$

The function $\mathcal{E}$ takes as input a public message number $N \in \{0,1\}^n$, associated data $A \in \{0,1\}^*$, and a message $M \in \{0,1\}^+$. It returns a ciphertext $C \in \{0,1\}^+$, where $|C| = |M|$, and tag $T \in \{0,1\}^\tau$: $(C,T) \leftarrow \mathcal{E}(N,A,M)$. The decryption function $\mathcal{D}$ takes as input a public message number $N \in \{0,1\}^n$, associated data $A \in \{0,1\}^*$, ciphertext $C \in \{0,1\}^+$, and tag $T \in \{0,1\}^\tau$. The algorithm $\mathcal{D}$ outputs $M \in \{0,1\}^+$ if the tag is correct and $\perp$ otherwise, which we denote as $M/\perp \leftarrow \mathcal{D}(N,A,C,T)$.

We first define AES-COPA for the case where the length of message M is a positive multiple of 16 bytes (128 bits). In Sect. 1.4 we explain how AES-COPA deals with fractional data. Let $L \stackrel{\text{def}}{=} E_k(0)$. The encryption and decryption procedures of AES-COPA on a message $M[1]M[2]\cdots M[d]$ of d 128-bit blocks and on a ciphertext and tag pair $(C[1]C[2]\cdots C[d], T)$ are then defined as:

AES-COPA-ENCRYPT:
if $\big|M[d]\big| = n$ **then**
  $V \leftarrow \text{PMAC1}'(A\|N)$
  $(C,S) \leftarrow \text{ENCRYPT}(V,M)$
  $\Sigma \leftarrow M[1] \oplus M[2] \oplus \cdots \oplus M[d]$
  $T \leftarrow E_k\big(E_k(\Sigma \oplus 2^{d-1}3^2L) \oplus S\big) \oplus 2^{d-1}7L$
  Output $(C, \lfloor T \rfloor_\tau)$
end if

AES-COPA-DECRYPT:
if $\big|C[d]\big| = n$ **then**
  $V \leftarrow \text{PMAC1}'(A\|N)$
  $(M,S) \leftarrow \text{DECRYPT}(V,C)$
  $\Sigma \leftarrow M[1] \oplus M[2] \oplus \cdots \oplus M[d]$
  if $\lfloor E_k\big(S \oplus E_k(\Sigma \oplus 2^{d-1}3^2L)\big) \oplus 2^{d-1}7L \rfloor_\tau = T$ **then**
    Output M
  else
    Output $\perp$
  end if
end if

where the subroutines are defined as:

PMAC1′(X):
$X[1]X[2]\cdots X[x] \leftarrow X$
$\Delta_0 \leftarrow 3^3 L$, $U \leftarrow 0$
for $i = 1, \ldots, x-1$ **do**
  $U \leftarrow U \oplus E_k\big(X[i] \oplus \Delta_0\big)$
  $\Delta_0 \leftarrow 2\Delta_0$
end for
if $\big|X[x]\big| = n$ **then**
  $V \leftarrow E_k\big(U \oplus X[x] \oplus 3\Delta_0\big)$
else
  $V \leftarrow E_k\big(U \oplus X[x]\|10^* \oplus 3^2\Delta_0\big)$
end if
Output V

Encrypt(V, M):
$V[0] \leftarrow V \oplus L, \Delta_0 \leftarrow 3L, \Delta_1 \leftarrow 2L$
for $i = 1, \ldots, d$ **do**
  $V[i] \leftarrow E_k\big(M[i] \oplus \Delta_0\big) \oplus V[i-1]$
  $C[i] \leftarrow E_k\big(V[i]\big) \oplus \Delta_1$
  $\Delta_0 \leftarrow 2\Delta_0, \Delta_1 \leftarrow 2\Delta_1$
end for
$C \leftarrow C[1]C[2]\cdots C[d]$, $S \leftarrow V[d]$
Output (C, S)

Decrypt(V, C):
$V[0] \leftarrow V \oplus L, \Delta_0 \leftarrow 3L, \Delta_1 \leftarrow 2L$
for $i = 1, \ldots, d$ **do**
  $V[i] \leftarrow E_k^{-1}\big(C[i] \oplus \Delta_1\big)$
  $M[i] \leftarrow E_k^{-1}\big(V[i] \oplus V[i-1]\big) \oplus \Delta_0$
  $\Delta_0 \leftarrow 2\Delta_0, \Delta_1 \leftarrow 2\Delta_1$
end for
$M \leftarrow M[1]M[2]\cdots M[d]$, $S \leftarrow V[d]$
Output (M, S)

See Figures 1 and 2 for a pictorial description. In PMAC1′, the last block $X[x]$ is padded (if not a multiple of n bits) by a one and as many zeroes as necessary to obtain a multiple of the block size n. Here, the block "$X[x]10^*$" replaces the block "$X[x]$" if $X[x]$ itself is not n bits.

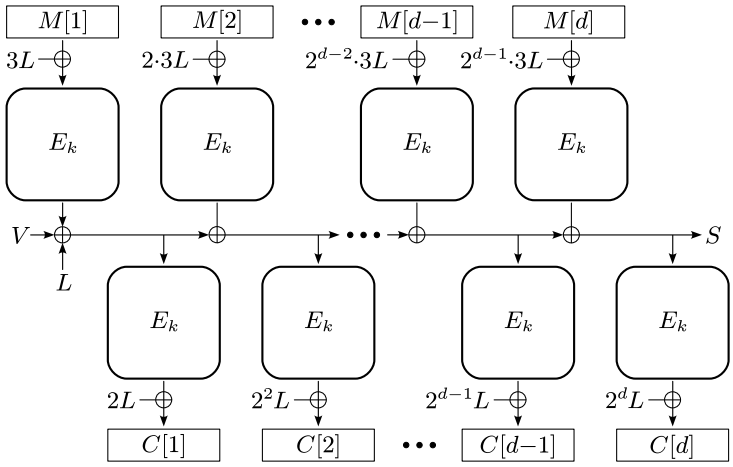
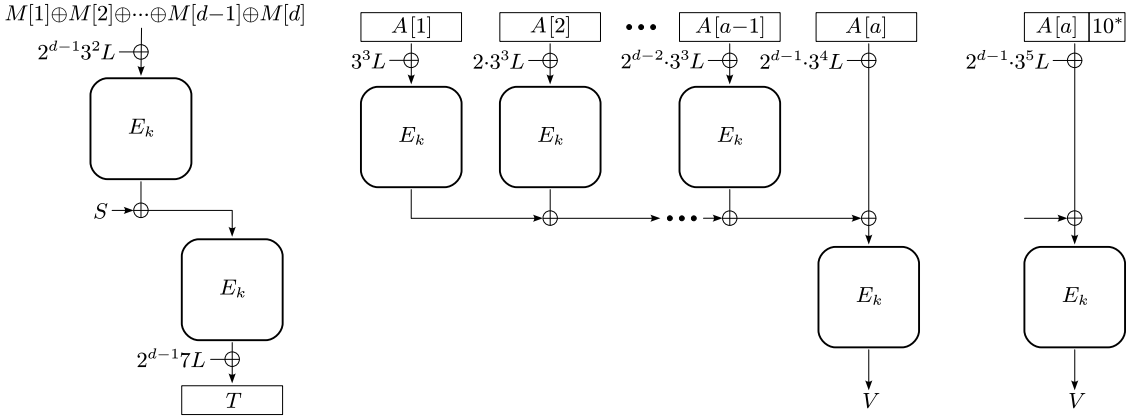


Figure 1: COPA plaintext processing.



(a) Tag generation (b) Processing of associated data

Figure 2: Tag generation and processing of associated data.

1.4 Fractional Data

We explain how to extend AES-COPA to accept "fractional" messages M. Here the length $|M|$ is not necessarily a positive multiple of the block size n. Note that simply using 10^* padding to M would result in ciphertext expansion. The methods described here avoid such expansion with minimal loss of efficiency.

1.4.1 Tag Splitting for $|M| < n$

For the case $|M| = s < n$, we define AES-COPA as:

AES-COPA-Encrypt:

if $d = 1$ and $\big|M[d]\big| = s < n$ **then**
$V \leftarrow \mathrm{PMAC1}'(A\|N)$
$S \leftarrow V \oplus 3^6L \oplus E_k\big(M\|10^* \oplus 3^7L\big)$
$C' \leftarrow E_k(S) \oplus 2 \cdot 3^6L$
$T' \leftarrow E_k\big(E_k(M\|10^* \oplus 3^8L) \oplus S\big) \oplus 3^67L$
$C \leftarrow \lceil C' \rceil_s$ (leftmost s bits)
$T \leftarrow \lfloor C' \rfloor_{n-s} \lceil T' \rceil_s$
Output (C, T)
end if

AES-COPA-DECRYPT:
if $d = 1$ and $\big|C[d]\big| = s < n$ **then**
  $C' \leftarrow C\lceil T \rceil_{n-s}$
  $V \leftarrow \text{PMAC1}'(A \| N)$
  $S \leftarrow E_k^{-1}(2 \cdot 3^6 \oplus C')$
  $M10^* \leftarrow 3^7 L \oplus E_k^{-1}\big(V \oplus 3^6 L \oplus S\big)$
  $T^* \leftarrow E_k\big(S \oplus E_k(M10^* \oplus 3^8 L)\big) \oplus 3^6 7L$
  if $|M| = s$ and $\lfloor T \rfloor_s = \lceil T^* \rceil_s$ **then**
    Output M
  else
    Output $\perp$
  end if
end if

Note that the integrity relies on the 10^* padding as well as on the "partial" tag $\lceil T' \rceil_s$.

1.4.2 XLS for $|M| > n$

Our solution relies on the XLS construction [5] of VIL tweakable ciphers. Let M be a message of at least n bits. Divide it into blocks as $M[1]M[2] \cdots M[d-1]M[d] \leftarrow M$, and assume that we have ($d \geq 2$ and) $1 \leq \big|M[d]\big| \leq n - 1$. Then AES-COPA for this case is defined as:

AES-COPA-ENCRYPT:
if $d \geq 2$ and $\big|M[d]\big| < n$ **then**
  $V \leftarrow \text{PMAC1}'(A \| N)$
  $(C', S') \leftarrow \text{ENCRYPT}(V, M[1]M[2] \cdots M[d-1])$
  $\Sigma' \leftarrow M[1] \oplus M[2] \oplus \cdots \oplus M[d-1]$
  $T' \leftarrow E_k\big(E_k(\Sigma' \oplus 2^{d-2}3^2 L) \oplus S'\big) \oplus 2^{d-2}7L$
  $C[d]T \leftarrow \text{XLS}_d(M[d]T')$
  $C \leftarrow C'C[d]$
  Output (C, T)
end if

AES-COPA-DECRYPT:
if $d \geq 2$ and $\left|C[d]\right| < n$ **then**
$V \leftarrow \text{PMAC1}'(A\|N)$
$(M', S') \leftarrow \text{DECRYPT}(V, C[1]C[2]\cdots C[d-1])$
$M[1]M[2]\cdots M[d-1] \leftarrow M'$
$\Sigma' \leftarrow M[1] \oplus M[2] \oplus \cdots \oplus M[d-1]$
$M[d]T' \leftarrow \text{XLS}_d^{-1}(C[d]T)$
if $S' \oplus E_k(\Sigma' \oplus 2^{d-2}3^2L) = E_k^{-1}(T' \oplus 2^{d-2}7L)$ **then**
Output $M[1]M[2]\cdots M[d-1]M[d]$
else
Output $\perp$
end if
end if

$\text{XLS}_d(X)$:
Replace the left n bits of X, LastFull, with $E_{d,2}$(LastFull)
Replace the last $2s$ bits of X, Last, with `mix`(Last)
Replace the $(|X| - 2s)$-th bit of X, b, with `flip`(b)
Replace the first l bits of X, First, with $E_{d,1}$(First)
Replace the $(|X| - 2s)$-th bit of X, b, with `flip`(b)
Replace the last $2s$ bits of X, Last, with `mix`(Last)
Replace the left n bits of X, LastFull, with $E_{d,2}$(LastFull)
Return X

Here $E_{d,1}$ is defined as $E_{d,1}(X) := E_k(X \oplus 2^{d-1}7^2L) \oplus 2^{d-1}7^2L$ and $E_{d,2}$ as $E_{d,2}(X) := E_k(X \oplus 2^{d-1}3 \cdot 7^2L) \oplus 2^{d-1}3 \cdot 7^2L$. The function `mix` is defined as

$$\texttt{mix}(AB) := (A \oplus \texttt{rol}(A \oplus B))\|(B \oplus \texttt{rol}(A \oplus B)),$$

where A and B are equal-length strings and $\texttt{rol}(X)$ represents left circular bit-rotation, that is, for any string X of length s we have $\texttt{rol}(X) = X_2X_3\cdots X_sX_1$. The function `flip` is simply bit flipping; i.e. $\texttt{flip}(b) := b \oplus 1$. Clearly XLS_d is invertible, and we write XLS_d^{-1} to denote its inverse.

2 Security Claims

In this section we specify the security levels with respect to the recommended key length of 16 bytes, tag length of 16 bytes, nonce length of 16 bytes. For these parameters AES-COPA achieves the following security levels (in $\log_2$ of number of AES calls):

	AES-COPA
confidentiality for the plaintext	64
integrity for the plaintext	64
integrity for the associated data	64
integrity for the public message number	64
security against key recovery	128
security against tag guessing	128

The security levels of AES-COPA correspond to the birthday bound security on the block size of AES[1] (see also Sect. 3). The security levels apply *both* in the cases when nonces are unique values (full security) and also when nonces are reused (full security up to common prefix, the maximum attainable for single pass schemes). We refer to [1] for the technicalities. Security against key recovery is 128 bits and security against tag guessing is 128 bits.

3 Security Analysis

AES is believed not to be distinguishable from a permutation drawn uniformly at random. In [1] we show that under this assumption (AES is a strong pseudo-random permutation SPRP) AES-COPA cannot be distinguished from an ideal authenticated encryption scheme in up to about 2^{64} AES calls. That is, AES-COPA is confidentiality secure (in the sense of indistinguishability from an online permutation) against chosen-plaintext (CPA) attacks and integrity secure against forgery up to approximately 2^{64} AES calls. Most importantly, and in contrast with the majority of existing authenticated encryption schemes, AES-COPA security proofs hold for stronger nonce-repeating attackers and is hence secure against nonce misuse.

More precisely, in [1] we prove the nonce misuse security of AES-COPA in the standard model under the SPRP assumption on AES against an AE distinguisher up to the bound $\frac{39(\sigma+q)^2}{2^n} + \frac{(l+2)(q-1)}{2^n} + \frac{2q}{2^n}$, where q are the AES-COPA queries of total length at most σ blocks and each of length at most l blocks.

This results means that under the same key the amount of ciphertext should not exceed 2^{64} blocks. We note, that many existing AES-128 based authenticated encryption schemes with the same security levels conservatively limit the total amount of plaintext and associated data blocks under a fixed key to at most 2^{48} blocks (2^{52} bytes).

When a ciphertext decrypts to $\perp$ the implementor needs to ensure that no information beyond this fact is leaked to the adversary. An adversary can always produce a valid tag T with probability $2^{-|T|}$. It is recommended that applications take care of producing ciphertexts with a single tag size under the same key to avoid potential security degradation.

[1]We clarify that our security results are of the same order as AES based authenticated encryption schemes, such as AES-GCM and AES-OCB

We clarify that our security model does not encompass timing and power consumption attacks.

4 Features of AES-COPA

Online. AES-COPA is based on an online cipher and is designed to allow for online processing for both encryption and decryption, processing data on-the-fly as it arrives.

Nonce misuse resistance. AES-COPA is designed to maintain security when the nonce is reused. More specifically, it achieves the maximal possible security against nonce reuse for an online authenticated encryption scheme [3], meaning that when two plaintexts are encrypted using the same associated data A and nonce N, the adversary can only determine the length of the common plaintext prefix of the two messages, since these will have the same corresponding ciphertext blocks.

Efficiency. AES-COPA is designed to allow high-performance implementations in both software and hardware.

- Parallelizability: AES-COPA can fully parallelize the execution of AES calls for subsequent message blocks once the respective masks have been computed. The same holds for the second layer of AES calls. For messages longer than one block, this significantly increases the performance. For shorter messages, the available parallelism in AES-COPA can be exploited by processing multiple messages simultaneously. Both approaches naturally extend to the case where multiple cores are available.
- AES-COPA is designed for efficiency for both short and long messages. Besides the AES key schedule, the overhead for short messages basically only amounts to two AES calls for the tag generation. On Intel's recent Haswell microarchitecture, AES-COPA achieves a performance of up to 1.29 cycles/byte (cpb) for longer messages (2048 bytes) and 1.44 cpb for shorter messages (128 bytes).
- Key agility (computational cost under distinct keys): AES-COPA requires one extra AES call every time a new key is used.
- Nonce agility (computational cost under use of nonces): Since nonces are appended to the associated data A, changing the nonce requires one extra AES call.
- Ability to efficiently preprocess A: Associated data (excluding the nonce) can be preprocessed independently of the message or the value of the nonce.
- Ability to efficiently preprocess plaintext: In the same way, the message (or parts thereof) can be preprocessed without seeing A in the first layer of AES calls. This also applies if the nonce is used.

Encryption and decryption do not require both AES and its inverse. AES-COPA encryption requires only forward AES operations while decryption requires only inverse AES operations if there is no associated data A, the subkey L is stored, and no tag truncation takes place (i.e., the recommended tag size is used).

Combination of well-known techniques. AES-COPA relies on the design principles of PMAC to achieve integrity of both the associated data and the plaintext. For the confidentiality we use masking techniques which instantiate an XEX tweakable block cipher on top of the basic underlying and well-understood AES block cipher.

4.1 Comparison to AES-GCM

Security against stronger adversaries. Compared to AES-GCM, AES-COPA guarantees security against a stronger adversary in the nonce misuse setting. In the nonce misuse setting the security of AES-GCM fails completely.

Performance. On platforms without fast multiplication instructions (including Sandy/Ivy Bridge), AES-COPA is faster than AES-GCM for all message lengths. For instance, on Sandy Bridge, AES-COPA runs at up to 1.70 cpb, while AES-GCM requires around 2.53 cpb. Even on Intel's Haswell platform with its improved multiplication instructions, AES-COPA offers comparable performance to AES-GCM in the case where multiple messages are processed in parallel, both for short and long messages [2]. Note that this is a very common scenario both in client-server protocols and in network packet processing. More generally, on platforms where AES computation is faster than GF(2^{128}) multiplication AES-COPA is faster than AES-GCM. This especially implies better implementation characteristics for platforms such as ARM or 8-bit microcontrollers, where AES can be implemented very efficiently.

Absence of weak keys as in polynomial hashing. There are no weak keys for AES-COPA like those existing for AES-GCM [4].

No universal hashing required. Unlike AES-GCM, AES-COPA does not require full GF(2^{128}) multiplications.

5 Design Rationale

AES-COPA has been designed to allow for high performance in parallel environments and to maintain security even if nonce is reused. Among other platforms, AES-COPA is well-suited for both Intel's AES-NI and high-performance hardware, being based on AES and small-constant multiplications in the finite field. A PMAC-like construction accounts for integrity and a XEX-type construction is used for assuring confidentiality in AES-COPA. AES is well-studied and wide-spread (including implementations and countermeasures

against side-channel analysis), being, thus, the natural choice for the underlying block cipher. The designers have not hidden any weaknesses in AES-COPA.

6 Intellectual Property

AES-COPA uses a PMAC-like construction for achieving integrity. According to its designers, one or more patents related to PMAC are pending. Depending on the breadth of the claims in these patents, this part of AES-COPA could be covered by them.

The submitters however explicitly state that AES-COPA itself will not be patented. If any of this information changes, the submitters will promptly (and within at most one month) announce these changes on the `crypto-competitions` mailing list.

7 Consent

The submitters hereby consent to all decisions of the CAESAR selection committee regarding the selection or non-selection of this submission as a second-round candidate, a third-round candidate, a finalist, a member of the final portfolio, or any other designation provided by the committee. The submitters understand that the committee will not comment on the algorithms, except that for each selected algorithm the committee will simply cite the previously published analyses that led to the selection of the algorithm. The submitters understand that the selection of some algorithms is not a negative comment regarding other algorithms, and that an excellent algorithm might fail to be selected simply because not enough analysis was available at the time of the committee decision. The submitters acknowledge that the committee decisions reflect the collective expert judgments of the committee members and are not subject to appeal. The submitters understand that if they disagree with published analyses then they are expected to promptly and publicly respond to those analyses, not to wait for subsequent committee decisions. The submitters understand that this statement is required as a condition of consideration of this submission by the CAESAR selection committee.

References

[1] 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)

[2] Bogdanov, A., Lauridsen, M.M., Tischhauser, E.: AES-Based Authenticated Encryption Modes in Parallel High-Performance Software. Cryptology ePrint Archive, Report 2014/186 (2014), `http://eprint.iacr.org/`

[3] 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)

[4] Procter, G., Cid, C.: On Weak Keys and Forgery Attacks against Polynomial-based MAC Schemes. In: FSE 2013 (2013)

[5] 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)