General Classification of the Authenticated Encryption Schemes for the CAESAR Competition

Farzaneh Abed, Christian Forler, and Stefan Lucks

Bauhaus-Universität Weimar
`<firstname>.<lastname>@uni-weimar.de`

Abstract. An Authenticated encryption scheme is a scheme which provides privacy and integrity by using a secret key. In 2013, CAESAR (the "Competition for Authenticated Encryption: Security, Applicability, and Robustness") was co-founded by NIST and Dan Bernstein with the aim of finding authenticated encryption schemes that offer advantages over AES-GCM and are suitable for widespread adoption. The first round started with 57 candidates in March 2014; and nine of these first-round candidates where broken and withdrawn from the competition. The remaining 48 candidates went through an intense process of review, analysis and comparison. While the cryptographic community benefits greatly from the manifold different submission designs, their sheer number implies a challenging amount of study. This paper provides an easy-to-grasp overview over functional aspects, security parameters, and robustness offerings by the CAESAR candidates, clustered by their underlying designs (block-cipher-, stream-cipher-, permutation-/sponge-, compression-function-based, dedicated). After intensive review and analysis of all 48 candidates by the community, the CAESAR committee selected only 30 candidates for the second round. The announcement for the third round candidates was made on 15th August 2016 and 15 candidates were chosen for the third round.

Keywords: authenticated encryption, CAESAR competition, symmetric cryptography.

1 Introduction

Confidential messages that shall be submitted over an insecure channel usually require protection of not only their privacy or confidentiality, but also of the authenticity or integrity of their respective sender. AE schemes are key-based cryptographic algorithms that try to provide both goals simultaneously.

Data privacy is for preventing an adversary to reveal any information except the length of a message sent from the sender to the receiver. So, notion of privacy keeps information about the message secret from all but only authorized parties to see it. And notion of integrity or authenticity confirms the original source of information, ensuring that original information has not been modified by unauthorized or unknown parties. To have authenticated encryption, we need both of these security notions.

The notion of AE was introduced by the seminal work by Bellare and Namprempre around 2000 [25,26], and further evolved during the past decade [121,124,127]. One of the main enhancements was the consideration for "associated data", which are not confidential, but must be included into the authentication. As an example, consider a network packet. Its destination address, which is part of the header, is public information needed to route the packet. But changing the header, e.g., by changing the destination address, could actually change the meaning of the encrypted message. Thus, both header and message must be authenticated.

The natural way to design an AE scheme is "generic composition": Pick a secure encryption scheme and a secure message authentication code (MAC) under two independent keys and use both of them. As natural as this approach is, the resulting scheme can theoretically turn out to be insecure. More precisely, there are three natural ways to generically combine a MAC and an encryption scheme: MAC-then-encrypt,

Encrypt-and-MAC, and Encrypt-then-MAC, and only the third one is guaranteed to be secure if the encryption scheme and the MAC are secure [25,26]. This approach is natural and easy to analyse. But it is also a bit slow, it requires two independent keys for encryption and authentication, and it is not robust to implementation errors (see, e.g., [59]).

The alternative to AE by generic composition, and the topic of the current paper, are dedicated AE schemes. Soon after 2000, authors did propose dedicated AE schemes based on a block cipher as the underlying primitive [65,86,88,126]. Some of them, such as, e.g., CCM [56], are "two-pass modes", which mimic generic composition, except that they avoid the usage of two independent keys.

The "Galois Counter Mode" (GCM) [100] is a two-pass mode based on a block cipher. GCM employs encryption scheme (a block cipher in "counter mode") and a polynomial hash function, using Galois field arithmetic. GCM has been so successful in practice, that it has been chosen as the reference system for CAESAR (see below).[1]

Another class block-cipher-based AE schemes are "single-pass modes", which do not mimic generic composition and are, at least potentially, more efficient. These include XCBC [65] and OCB [126]. Their usage appears to be hindered by the patent situation.

Block ciphers are traditionally the workhorses of Symmetric Cryptography. Nevertheless, there are other approaches to AE schemes not based on block ciphers, but based on either

- a keyless permutation [33],
- or a stream cipher [5],
- or a hash or compression function [62].

Also, there are dedidated AE schemes without an underlying primitive [38,60,145]; in some sense, these schemes can be seen as primitives of their own right.

Despite the variety of available designs, at the beginning of 2013, a large amount of SSL/TLS servers still employ RC4 [119]—most likely as a backup strategy against attacks [7,55], and perhaps also due to performance reasons. At the workshop on Fast Software Encryption (FSE) 2013 [27], Bernstein outlined the most obvious needs on AE schemes: Can one construct AE schemes that offer a higher level of security than GCM with similar performance; or such that are faster than GCM with a similar level of security. Moreover, the community derived many further desirable features from practical needs: Can AE schemes be designed to be fast in hard- and software, to detect forgery attempts fast, to provide robustness against nonce misuse or against leakage of invalid plaintexts, etc. So, there still seems to be an enormous gap that motivates a concentrated research on novel designs.

CAESAR (the Competition for Authenticated Encryption: Security, Applicability, and Robustness) contest aims at filling this gap for AE. At January 2013, the CAESAR call for submissions asked for AE schemes that should "*(1) offer advantages over AES-GCM and (2) are suitable for widespread adoption*" [28]. The call was responded by 57 submissions in total – many of which proposed several recommendations for their primitives, or even multiple different instantiations. While analysts and designers can learn lots from the heterogeneous field of candidates, their sheer number implies a challenging amount of study for submitters and analysts to keep track of every scheme's individual advantages and drawbacks.

Contribution. In this paper, we provide a comprehensive overview on the first-round CAESAR candidates, inspired by the preliminary summary by Bart Preneel at the Dagstuhl Seminar 14021 [14].

[1] Note that [118,132] identified considerable groups of weak keys.

We propose an intuitive classification of the candidates according to their design approaches (block-cipher-based, stream-cipher-based, permutation-/sponge-based, compression-function-based, dedicated). Our goal is to provide easy-to-grasp tables to compare individual functional features (parallelizeability, onlineness, inverse-freeness, support for intermediate tags, and incrementality) for CAESAR candidates, to compare their security parameters (for privacy and integrity), as well as their robustness offering (nonce- and decryption-misuse). This overview paper provides a lot of useful information for the cryptanalyst community as well as CAESAR committee. Cryptanalyst can choose their favourite candidates based on primitive, security parameters or up-to-date external analysis of candidates which are all classified in this paper. We make committee member work much easier to know the current status of the candidates and their analysis. The AE Zoo [4] also holds very useful information.

Note that we consider only recommended parameter sets for those candidates that have not been withdrawn from the competition, which is the case for 48 out of the 57 submissions. At the time, we exclude AES-COBRA [11], Calico [139], CBEAM [130], FASER [44], HKC [73], Marble [69], McMambo [92], PAES [153], and PANDA [154]. Furthermore, we explicitly do not consider performance measures since the SUPERCOP framework and website [29] provide the better platform for this purpose.

Outline. The remainder of this paper is organized as follows: Section 2 defines preliminaries. Section 3 explains the underlying primitives, and lists the functional characteristics of authenticated encryption schemes. Section 4 briefly recalls the relevant security and robustness notions and criteria. The general overview of attacks on candidates is explained in Section 5. In the last section, Section 6, the schemes are compared in a table for functional features and security parameters, .

2 Preliminaries

Nonces. An encryption scheme – authenticated or not – shall not even leak the information that two identical plaintexts have been encrypted twice. Thus, encryption schemes must be either state-based or probabilistic. On the other hand, it is desirable to define the encryption as a single deterministic encryption function. This lead to the introduction of nonce-based encryption [75]. A nonce is an auxiliary input, representing a number used only once. The user (i.e., the sender of a message) is responsible for the nonce to be unique. The user can use state-based nonces (e.g., a counter which is incremented with every message), or choose random nonces, in which case the uniqueness should be a statistical expectation. While our notation assumes explicit nonces, some AE schemes define the nonces implicitly, as a part of the associated data. A nonce can be public and known to the adversary, or it can be secret. The CAESAR call for submissions supports the nonce to be split into a public and a secret message number (PNM, SNM).

Authenticated Encryption. Encryption is about providing privacy of data under a secret key. The sender may encrypt a confidential plaintext and transmit the ciphertext, the receiver may decrypt the ciphertext, returning the plaintext. Authenticated encryption is both about privacy and authenticity of data under a secret key. The main difference is that the decryption may return the special symbol $\perp$ instead of the plaintext, indicating a possible forgery attempt. In practice, the authenticity of a plaintext often depends on some non-confidential context information, the "associated data" or "header".

Formal Definition: Authenticated Encryption with Associated Data (AEAD). An authenticated encryption scheme with associated data, AEAD [122] is a tuple $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ with an encryption algorithm $\mathcal{E}_K(H,N,M)$ and a decryption algorithm $\mathcal{D}_K(H,N,C,T)$. $K \in \mathcal{K}$ denotes the key, $H \in \mathcal{H}$ the associated data (or header), $N \in \mathcal{N}$ the nonce[2], $M \in \mathcal{M}$ the message, $T \in \mathcal{T}$ the authentication tag, and $C \in \mathcal{C}$ the ciphertext, where $\mathcal{K} \subseteq \{0,1\}^k$, $\mathcal{H}, \mathcal{M}, \mathcal{C} \subseteq \{0,1\}^*$, and $N \in \{0,1\}^n$, $\mathcal{T} \subseteq \{0,1\}^t$ denote the key, header, message, ciphertext, nonce and tag space, respectively, with $k, t > 1$. We write

$$\begin{aligned} \mathcal{E}: &\quad \mathcal{K} \times \mathcal{H} \times \mathcal{N} \times \mathcal{M} \quad \to \mathcal{C} \times \mathcal{T}, \\ \mathcal{D}: &\mathcal{K} \times \mathcal{H} \times \mathcal{N} \times \mathcal{C} \times \mathcal{T} \to \mathcal{M} \cup \{\perp\}, \end{aligned}$$

for the encryption function $\mathcal{E}$ and the decryption function $\mathcal{D}$.

Thus,

$$\mathcal{E}_K(H,N,M) = (C,T)$$

is the output of encrypting the message M using the header H and the nonce N under the key K. Note that (C,T) is just a single string. While most AE schemes support the division of (C,T) into a core ciphertext C and an authentication tag T, a few of them, such as AEZ, do not support this distinction.

Decryption is performed similarly to encryption:

$$\mathcal{D}_K(H,N,C,T) = \begin{cases} M \text{ if } (C,T) \text{ is valid} \\ \perp \text{ else} \end{cases}$$

The scheme is correct if

$$\mathcal{D}_K(H,N,\mathcal{E}_K(H,N,M)) = M$$

holds for each quadruple (K,H,N,M).

Separated AEAD Scheme. Decryption, as defined above, can *either* output a message $M \in \mathcal{M}$, *or* the special symbol $\perp$ to indicate a forgery attempt being discovered. This definition does not allow to model a scenario where the adversary can find out anything about a would-be message $M \in \mathcal{M}$, when decryption eventually discovers a forgery attempt. Separated AEAD allows to model such scenarios.

A separated authenticated encryption scheme with associated data [12] is a triple $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D}, \mathcal{V})$ with an encryption algorithm $\mathcal{E}_K(H,N,M)$, a decryption algorithm $\mathcal{D}_K(H,N,C,T)$, and a verification algorithm $\mathcal{V}_K(H,N,C,T)$. $K \in \mathcal{K}$ denotes the key, $H \in \mathcal{H}$ the associated data (or header), $N \in \mathcal{N}$ the nonce, $M \in \mathcal{M}$ the message, $T \in \mathcal{T}$ the authentication tag, and $C \in \mathcal{C}$ the ciphertext, where $\mathcal{K} \subseteq \{0,1\}^k$, $\mathcal{H}, \mathcal{M}, \mathcal{C} \subseteq \{0,1\}^*$, $N \in \{0,1\}^n$ and $\mathcal{T} \subseteq \{0,1\}^t$ denote the key, header, message, ciphertext, nonce and tag space, respectively, with $k, t > 1$. We write

$$\begin{aligned} \mathcal{E}: &\quad \mathcal{K} \times \mathcal{H} \times \mathcal{N} \times \mathcal{M} \quad \to \mathcal{C} \times \mathcal{T}, \\ \mathcal{D}: &\mathcal{K} \times \mathcal{H} \times \mathcal{N} \times \mathcal{C} \times \mathcal{T} \to \mathcal{M}, \\ \mathcal{V}: &\mathcal{K} \times \mathcal{H} \times \mathcal{N} \times \mathcal{C} \times \mathcal{T} \to \{\top, \perp\}, \end{aligned}$$

for encryption, decryption and verification. Unlike the standard definition for authenticated encryption, the separated decryption function

$$\mathcal{D}_K(H,N,C,T) = M$$

always returns a plaintext M, even when the authentication of (C,T) failed. In the case of an authentication failure, the verification function returns $\perp$, i.e.,

$$\mathcal{V}_K(H,N,C,T) = \perp.$$

An AEAD scheme $\Pi = (\mathcal{E}, \mathcal{D})$ is separated, if it can be rewritten as a separated AEAD. Otherwise, we call it **conventional**.

[2] Some times the nonce is explicit but sometimes it is implicitly part of the header.

On-line AE. Encryption – authenticated or not – can be defined such that the sender of a long plaintext can emit parts of the ciphertext before having read the entire plaintext (i.e., on-line), or such that the sender always must read the entire plaintext before it can start sending out the first bits of the ciphertext (this characterises an off-line scheme). Formally, on-line is based on the notion of an on-line cipher.

Let $\mathit{\Gamma} : \{0,1\}^k \times (\{0,1\}^n)^* \to (\{0,1\}^n)^*$ denote a keyed family of n-bit permutations, which takes a k-bit key K and a message M of an arbitrary number of n-bit blocks, and outputs a ciphertext C consisting of the same number of n-bit blocks as M. We call $\mathit{\Gamma}$ an ***on-line cipher*** iff the encryption of message block M_i, for all $i \in [1, |M|/n]$, depends only on the blocks $M_1, \ldots, M_i$.

An online cipher can not guarantee to behave like random permutation which is the critical condition for the secure cipher. The reason is that in online cipher, the encryption of current block of message M_i does not depend on the next block M_{i+1}. Hence, we call an on-line cipher $\mathit{\Gamma}$ secure if ciphertext reveals only the length of plaintext and the ***longest common prefix*** with previous messages, and no further information.

For integers $n, \ell, d \geq 1$, let $\mathcal{D}_n^d = (\{0,1\}^n)^d$ denote the set of all strings that consist of exactly d blocks of n bits each. Further, let $\mathcal{D}_n^* = \bigcup_{d \geq 0} \mathcal{D}_n^d$ denote the set which consists of all possible n-bit strings and $\mathcal{D}_{\ell,n} = \bigcup_{0 \leq d \leq \ell} \mathcal{D}_n^d$ the set of all possible strings which consist of 0 to ℓ n-bit blocks. For arbitrary $P \in \mathcal{D}_n^d$, let P_i denote the i-th block for all $i \in 1, \ldots, d$. For $P, R \in \mathcal{D}_n^*$, we define the *length of the longest common prefix of n-bit blocks*, Prefix of P and R [61] by

$$\mathsf{LLCP}_n(P, R) = \max_i \{\forall j \in 1, \ldots, i : \ P_j = R_j\}.$$

For a non-empty set $\mathcal{Q}$ of strings in D_n^*, we define

$$\mathsf{LLCP}_n(\mathcal{Q}, P) = \max_{q \in \mathcal{Q}} \{\mathsf{LLCP}_n(q, P)\}.$$

For any two ℓ-block inputs M and M' with $M \neq M'$, that share an exactly m-block common prefix $M_1 \mid\mid \ldots \mid\mid M_m$, the corresponding outputs $C = P(M)$ and $C' = P(M')$ satisfy $C_i = C'_i$ for all $i \in [1, m]$ and $m \leq \ell$, where P denotes an on-line permutation. However, it applies that $C_{m+1} \neq C'_{m+1}$ and all further blocks C_i and C'_i, with $i \in [m+2, \ell]$, to be *independent*. This behaviour is defined by ***on-line permutations***.

Let $F_i : (\{0,1\}^n)^i \to \{0,1\}^n$ be a family of indexed n-bit permutations, i.e., for a fixed $j \in (\{0,1\}^n)^{i-1}$ it applies that $F_i(j, \cdot)$ is a permutation. We define an n-bit on-line permutation $P : (\{0,1\}^n)^\ell \to (\{0,1\}^n)^\ell$ as a composition of ℓ permutations:

$$F(M_1, \ldots M_\ell) = \Big(F_1(M_1), F_2(M_1, M_2), \cdots F_\ell(M_1, \ldots M_\ell)\Big).$$

An ℓ-block message $M = (M_1, \ldots, M_\ell)$ is mapped to an ℓ-block output $C = (C_1, \ldots, C_\ell)$ by

$$C_i = F_i(M_1 \mid\mid \ldots \mid\mid M_{i-1}, M_i), \quad \forall i \in [1, \ell].$$

3 Underlying Constructions

This section briefly recalls the constructions that can be used as a primitive to construct an AE schemes. We consider the constructions which are used as a base of the CAESAR candidates.

Block Cipher. A block cipher is a keyed family of n-bit permutations $E : \{0,1\}^k \times \{0,1\}^n \rightarrow \{0,1\}^n$ which takes a k-bit key K and an n-bit message M and outputs an n-bit ciphertext C. We define $\texttt{Block}(k, n)$ as the set of all (k, n)-bit block ciphers for $n > 0$. For any $E \in \texttt{Block}(k, n)$ and a fixed key $K \in \{0,1\}^k$, the encryption of a message M is defined by $E_K(M)$, and the decryption is defined as the inverse function, i.e., $E_K^{-1}(M)$. For any key $K \in \{0,1\}^k$, it applies that $E_K^{-1}(E_K(M)) = M$.

AES-Based. The "Advanced Encryption Standard" (AES) [47] is the main standard block cipher in cryptography. Its block length is $n = 128$, there are many implementations available, and its security has been intensely analysed in almost two decades. Thus, many block cipher schemes actually use the AES. Starting with Intel's Westmere microarchitecture in 2011, current processors provide native AES instructions that allow fast constant-time encryption and decryption.

Stream Cipher. A stream cipher is a symmetric pseudo-random bit generator (PRBG) that takes a fixed-length secret key and generates a keystream of variable length [81]. This is done by adding a bit from a keystream to a plaintext. Stream ciphers are designed to be fast and small in size, and are mainly used for applications with low resources such as embedded devices. It can also used for encryption of Internet traffic such as RC4 stream cipher.

Like block ciphers, stream ciphers can be used as a core primitive in authenticated encryption scheme to achieve both confidentiality and integrity as long as the cipher is secure [60].

Key-Less Permutation. A key-less permutation is a bijective mapping on fixed-length strings. Permutations received a high level of attention during the SHA-3 competition[3] – last but not least due to its winner [31]. The most famous mode of operation for a keyless permutation is the sponge construction [30], which is an iterated function with variable-length in- and outputs from a permutation (or transformation) that itself operates on a fixed-length state. A sponge function can also be used as a stream cipher in this interface. The sponge construction operates on a state of b bits which comes from the bitrate r and the capacity c where $b = r + c$. Literally, the sponge is said to first *absorb* its inputs block by block before it processes and *squeezes* it out afterwards. Because of its arbitrarily input and output sizes, sponge construction can be used in many building primitives such as a stream cipher, hash function or message authentication code (MAC).

Duplex constructions are closely related to sponges [32]. Unlike sponge constructions, which are stateless between calls, a duplex accept calls that take an input string and return an output string which depends on all previous inputs. This property allows an efficient implementation of reseedable pseudo random generator and authenticated encryption scheme which require only one call to the permutation on each input block.

Hash Function/Compression Function. A hash function maps strings of arbitrary length to a fixed-length outputs or hash value. This implies that any changes of at least one bit of input should change the entire output. For cryptographic hash functions, it should be infeasible for an adversary to find a collision, preimage and second preimage. Some well-known algorithms like Message-Digest Algorithm 5 (MD5) or SHA-1 are the most used algorithms. Practical cryptographic hash functions are typically designed by iterating a "compression function" with constant-size inputs. Of course, an AEAD scheme can directly call the compression function, rather than indirectly, by calling the hash function.

[3] http://competitions.cr.yp.to/sha3.html

Dedicated. Some CAESAR candidates have a structure which is similar to that of Type-3 Feistel schemes [158]. Such schemes maintain a multi-block states $S_0, \ldots, S_n$, where each state is updated by feeding in one message block (e.g., $S_0 = S_0 \oplus M$) and updating each state with the result of its neighbour state block, processed by a round function: $S_i = S_i \oplus f(S_{i-1})$.

3.1 Underlying Modes and Masking Methods

Encryption Modes. An algorithm which uses an underlying primitive to provide security for confidentiality and authenticity is called mode of operation. Especially for block-cipher-based candidates, we will explicitly state which encryption mode(s) they inherit from. Moreover, we also list the underlying modes for some non-block-cipher-based candidates, when this is the case. The following modes adopted by the CAESAR candidates, and use the following acronyms:

CBC, CFB, CTR, ECB, OFB [114] It had long been known that a block cipher allone is not very useful. Even for plain encryption, one needs some "mode of operation", such as CBC, CFB, CTR, or OFB (Cipher Block Chaining, Cipher Feed-Back, Counter, Output Feed-Back). Note that the ECB mode (Electronic Code-Book) has been formally standardized, but is well-known to be insecure.

EME The Encrypt-Mix-Encrypt mode [71,70] turns an n-bit block cipher into a tweakable encryption scheme that acts on strings of mn bits, and it is paralleliazble. EME algorithm consists of two layers of ECB encryption and one non-linear mixing layer in between. EME is provably secure assuming the underlying primitive is strong pseudo random permutation (SPRP) secure.

iFeed the iFeed mode [157] can use a block cipher or a compression function as its underlying primitive. It is provably secure up to the birthday bound assuming the underlying primitive is secure.

JHAE The JH-based mode [80] has been inspired by the JH hash function mode. It is provably secure assuming that the permutation is ideal.

LEX The Leakage EXtraction mode [36] is defined for the AES. It is related to the OFB mode. But instead of using the ciphertext as a keystream, LEX uses values from intermediate rounds as a keystream.

OTR The offset Two-Round Feistel [104] uses a two-round Feistel permutation. OTR uses only the encryption algorithm for both the encryption and the decryption process. Encryption and decryption can be done in parallel. It is provably secure up to the birthday bound assuming the underlying primitive is a pseudorandom function (PRF) or a pseudorandom permutation (PRP).

PPAE Parallelizeable Permutation-based Authenticated Encryption [6] is based on a permutation and is fully parallelizeable. It provides security up to 128 bits and higher equal to the key length. Only forward permutation calls are used for both encryption and decryption.

SIV In general, the Synthetic Initialization Vector SIV-scheme [127] combines a MAC (with the requirement to behave pseudorandomly) with a nonce-based encryption scheme. All inputs go into the MAC, and the output of the MAC is used as the authentication tag. The specificality of SIV is that this authentication tag is used as the nonce for the encryption scheme. SIV decryption first uses the the tag as the nonce, to decrypt the ciphertext, then it will call the MAC to verify if it manages to generate the same tag.
SIV provides both nonce-based authenticated encryption as well as deterministic or nonce-less key wrapping.

XEX	XOR-encrypt-XOR (Even-Mansour) [123] is a tweakable mode of operation of block cipher. It is a simplest block cipher mode which XORs the plaintext with a key, applying a pseudo-random permutation to the result, and then XORing a same key to the permuted result to produce the final ciphertext.

Masking Methods. Many modern block-cipher-based schemes mask in- and outputs to the block cipher to prevent them from being under control of adversaries. From our finding, following approaches are used for the masking in the CAESAR candidates:

AX	Addition and XOR.
Doubling	XOR with a key-dependent variable that is incremented by doubling it in Galois Field [123] also known as finite field. The finite field with p^n elements is denoted by $GF(p^n)$, where the p is a prime number and n is a positive integer.
GFM	Multiplication with a key-dependent variable in Galois-Field.
AES	XORing an AES-processed chaining value [2].

3.2 Functional Characteristics

Different AE schemes would support different functional characteristics. In this section, we introduce relevant characteristics which we will use to classify CAESAR candidates.

Parallelizeable. We call an encryption operation parallelizable if the processing of the i-th input block does not depend on the output of processing the j-th block, for any $i \neq j$. Both encryption and decryption algorithms can be done in parallel. So, we regard parallelizeable encryption and decryption separately.

As a slightly weaker kind of this feature is *pipelineable* scheme. We call an AE scheme *pipelineable* if the encryption (and likewise the decryption) can be decomposed into operations $f \circ g$, such that the first operation $g(M_i)$ can be performed for the i-th block before the encryption of the previous blocks have finished.

Online. A cipher is online if the encryption of the i-th input block M_i depends only on the blocks $M_1, \ldots, M_{i-1}$ and only constant size-state is used from the processing of one block to the next. We call an AEAD scheme $\mathit{\Pi} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ online if $\mathcal{E}$ is an online cipher and $\mathcal{D}$ its inverse operation. Schemes that are not online are called **offline**.

Online Decryption. Note that the above definition for online AE considers only the encryption operation. If online encryption is desirable, why not online decryption?

For the answer, recall the formal definition for authenticated decryption:

$$\mathcal{D}_K(H, N, C, T) = \begin{cases} M \text{ if } (C,T) \text{ is valid} \\ \perp \text{ else} \end{cases}$$

Authenticated decryption is supposed to return *nothing* but $\perp$ in the case of an authentication failure. Since it is supposed to be infeasuble to forge unauthenticated ciphertexts for the adversary, the decryption process must read all its inputs (H, N, C, T) before it can accept them as authentic. Any implementation supporting online decryption would not make use of a desirable feature – rather, that implementation would be dangerously buggy.

In that sense, online decryption – even if functionally possible – is not a feature we will "advertise".

But note that we consider support for "intermediate tags" a relevant feature (see below). Intermediate tags can, indeed, serve to securely support some form of online decryption. If a long ciphertext $C = ((C_1, T_1), (C_2, T_2), \ldots,)$ can be split into authenticated fragments (C_i, T_i), one can securely release the partial decryption of $(C_1, T_1), \ldots (C_{i-1}, T_{i-1})$ before having read (C_i, T_i).

All CAESAR candidates supporting intermediate tags allow this kind of "online decryption".

Inverse-Free. AE schemes that employ only an encryption *or* decryption function can save precious memory and area resources. Wlog., we call an AE scheme inverse-free if it does not require either its underlying primitive's inverse operation, i.e., the block ciphers decryption function for block-cipher-based schemes, and the inverse permutation for permutation-based schemes.

Incremental Authenticated Encryption. AE schemes are frequently used to encrypt lots of data, wherein subsequent messages differ only by a fraction (e.g., a single block) from each other. An AE scheme is said to provide *incremental authenticated encryption*, if, given a previous authenticated ciphertext and tag (C, T) for a message M, encrypting and authenticating a message M' that differs from M only in a fraction, can be computed in proportional time and not the same as simply encrypting and authenticating a message M. Then recomputation of changed data will be significantly faster [24]. Incrementality is essentially a practical concern because it is measure of efficiency. Therefore, incremental scheme have this advantage over standard one specially for larger message size. In this paper, we assume that recomputation requires only the costs for processing the changed blocks and tag derivation.

Note that some schemes may provide this property under the requirement of reusing the nonce. We consider nonce misuse to be an *erroneous* usage which should not be encouraged to obtain a nice "feature". Hence, we denote scheme to provide incremental authenticated encryption only if the nonce is used only once and never is repeated.

Incremental Associated Data. This property is similar to incremental AE. Suppose, an intermediate result of a previous associated data processing is cached, and the current associated data changes only in a fraction. We say a scheme provides incremental associated data if only the changed blocks and a finalization step need to be recomputed [135].

Fixed Associated Data Reuse. Some applications use the same or slightly modified associated data values for subsequent messages [135]. Schemes that can cache and reuse the result of processing the associated data of the previous encrypted message may allow for a considerable speed-ups. We say that such schemes provide associated-data reuse. Note that this implies that the nonce is not part of or appended to the associated data.

Intermediate Tags. Intermediate tags [32] allow the receiver to detect early if parts of a decrypted message are invalid, which saves computations when authenticating large messages. Such information can be integrated easily into weak non-malleability of online cipher by adding well-formed redundancy, such as fixed constants or checksums [3]. Hence, we say that an AE scheme provides this property, if it is online and non-malleable ($\mathsf{OPRP\text{-}CCA}$-secure). By non-malleability, we mean that if adversary manipulates the i-th ciphertext block, then it cannot distinguish between the $(i+1), (i+2)$, ... ciphertext blocks of online cipher and random one. The scheme with support of intermediate tag can be well-suited for low-latency environments such as optical transport network (OTN), where messages usually contains of multiple TCP/IP packages with small integrated checksums.

4 Security

In this section, we give general overview of security notion for AE schemes and online AE schemes. The security aim of AE is to ensure both privacy and authenticity for encrypted messages at the same time. For our purpose, we consider some general security notion and also some advanced notions such as CCA3 security by Rogaway and Shrimpton [128] which includes IND-CPA and INT-CTXT notions. We also consider these notions for online ciphers: OCCA3 and OPRP-CPA [2]. We recall the notions in brief in the following subsection.

We consider an adversary $\mathcal{A}$ as an efficient Turing machine that interacts with a given set of oracles which operate as black boxes to $\mathcal{A}$, and is computationally bounded.

4.1 General Security Notions.

Pseudorandom Function (PRF). A pseudorandom function is a family of functions where the input-output behavior of a random instance of the family is 'computationally indistinguishable" from that of a random function. Means that an adversary that can only feed in inputs and get outputs, should be unable to tell whether the function in question is a random instance of the family in question or a random function. For some integers $k, l, L \geq 1$, we fix a family of functions $F : \mathcal{K} \times D \to R$, $(K = \mathcal{K}, D = \{0,1\}^l, R = \{0,1\}^L)$. There are two different ways in which $F_n : D \to R$ can be chosen, which we define it as a world. In the real world, F_n is a random instance of F, and in the random world, F_n is a random function with range R. The task of adversary is to discover which world it is in, meaning distinguishing between these two worlds. If it can guess with probability 1/2 or greater then it will win.

We define advantage as a measure of adversary's success to doing its job, such as determining which world it is in. The advantage can be any number between 0 and 1.

Definition 1 (PRF-Advantage). *Let $F : \mathcal{K} \times D \to R$ be a family of functions, and $\mathcal{A}$ an adversary which has access to oracle and returns a bit. The prf-advantage of $\mathcal{A}$ is given by*

$$\mathbf{Adv}_F^{\mathrm{PRF}}(\mathcal{A}) = \left|\Pr\left[Real_F^{\mathcal{A}} \Rightarrow 1\right] - \Pr\left[Rand_R^{\mathcal{A}} \Rightarrow 1\right]\right|.$$

Pseudorandom Permutation (PRP). Pseudorandom permutation is a family of functions $F : \mathcal{K} \times D \to D$ if the input and output behaviour of random instance of the family is indistinguishable from that of a random permutation. When F is a family of permutations, the adversary has access to the inverse oracle too. The scenario for the PRP is the same as the one in PRF.

Here we define advantage of adversary against Chosen-Plaintext-Attack or CPA and Chosen-Ciphertext-Attack or CCA.

Definition 2 (PRP-Advantage under CPA). *Let $F : \mathcal{K} \times D \to D$ be a family of functions, and $\mathcal{A}$ an adversary which has access to oracle and returns a bit. The PRP-CPA – advantage of $\mathcal{A}$ is given by*

$$\mathbf{Adv}_F^{\textsf{\textit{PRP-CPA}}}(\mathcal{A}) = \left|\Pr\left[Real_F^{\mathcal{A}} \Rightarrow 1\right] - \Pr\left[Perm_D^{\mathcal{A}} \Rightarrow 1\right]\right|.$$

Definition 3 (PRP-Advantage under CCA). *Let Let $F : \mathcal{K} \times D \to D$ be a family of functions, and $\mathcal{A}$ an adversary which has access to oracle and its inverse, and returns a bit. The PRP-CCA – advantage of $\mathcal{A}$ is given by*

$$\mathbf{Adv}_F^{\textsf{\textit{PRP-CCA}}}(\mathcal{A}) = \left|\Pr\left[Real_F^{\mathcal{A}} \Rightarrow 1\right] - \Pr\left[Perm_D^{\mathcal{A}} \Rightarrow 1\right]\right|.$$

In this scenario, the adversary has more power since it can also query from the decryption or inverse oracle. The family of F is a secure PRP under CCA and CPA if the advantage is small for all adversaries that are using practical amount of resources. Resources can be time complexity t of $\mathcal{A}$, number of queries q that $\mathcal{A}$ asks from the oracle, and the total length μ of all adversary's queries. It is worth to mention that PRP-CCA implies PRP-CPA security.

Definition 4 (IND-CPA-Security). *Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be an authenticated encryption scheme. Then, the IND-CPA-advantage of a computationally bounded adversary $\mathcal{A}$ for Π is defined as*

$$\mathbf{Adv}_{\Pi}^{\textsf{IND-CPA}}(\mathcal{A}) \leq \left| \Pr\left[K \xleftarrow{\$} \mathcal{K} : \mathcal{A}^{\mathcal{E}(\cdot,\cdot)} \Rightarrow 1\right] - \Pr\left[\mathcal{A}^{\$(\cdot,\cdot)} \Rightarrow 1\right] \right| .$$

We define $\mathbf{Adv}_{\Pi}^{\textsf{IND-CPA}}(q, \ell, t)$ as the maximum advantage over all IND-CPA-adversaries $\mathcal{A}$ on Π that run in time at most t, and make at most q queries of total length ℓ to the available oracles.

Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be an authenticated encryption scheme and $\mathcal{A}$ an IND-CPA adversary. The task of $\mathcal{A}$ is to distinguish the *real* world, where it is given oracle access to $E_K(\cdot,\cdot)$ under a secret key $K \in \mathcal{K}$, from the *random* world, where $\mathcal{A}$ has access to a random oracle $\$(\cdot,\cdot)$ which returns, consistent, random ciphertexts. If no such adversary $\mathcal{A}$ can perform significant better than random guessing, then, Π protects the privacy of encrypted messages [25].

Definition 5 (INT-CTXT-Security). *Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be an authenticated encryption scheme. Then, the INT-CTXT-advantage of a computationally bounded adversary $\mathcal{A}$ for Π is defined as*

$$\mathbf{Adv}_{\Pi}^{\textsf{INT-CTXT}}(\mathcal{A}) \leq \Pr\left[K \xleftarrow{\$} \mathcal{K} : \mathcal{A}^{\mathcal{E}(\cdot,\cdot),\mathcal{D}(\cdot,\cdot)} \Rightarrow \textit{ forges}\right]$$

We define $\mathbf{Adv}_{\Pi}^{\textsf{INT-CTXT}}(q, \ell, t)$ as the maximum advantage over all INT-CTXT-adversaries $\mathcal{A}$ on Π that run in time at most t, and make at most q queries of total length ℓ to the available oracles [25].

For the definitions and security notions regarding online ciphers, please see Bellare et al. [22].

Definition 6 (CCA3-Security). *Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be an authenticated encryption scheme. Then, the CCA3-advantage of a computationally bounded adversary $\mathcal{A}$ is defined as*

$$\mathbf{Adv}_{\Pi}^{\textsf{CCA3}}(\mathcal{A}) = \left| \Pr\left[K \xleftarrow{\$} \mathcal{K} : \mathcal{A}^{\mathcal{E}_K(\cdot,\cdot),\mathcal{D}_K(\cdot,\cdot,\cdot)} \Rightarrow 1\right] - \Pr\left[\mathcal{A}^{\$(\cdot,\cdot),\perp(\cdot,\cdot,\cdot)} \Rightarrow 1\right] \right| .$$

The CCA3 notion [129] states that $\mathcal{A}$ has access to an oracle $\mathcal{O}$, which provides $\mathcal{A}$ with an encryption and a decryption functions. At the beginning, $\mathcal{O}$ tosses a fair coin; depending on the result of the coin toss, $\mathcal{O}$ uses the *real* encryption $\mathcal{E}_K(\cdot,\cdot)$ and decryption $\mathcal{D}_K(\cdot,\cdot,\cdot)$ functions, or a *random* function $\$(\cdot,\cdot)$ for the encryption and a $\perp$ function for $\perp(\cdot,\cdot,\cdot)$, which returns $\perp$ on every input, for the decryption queries of $\mathcal{A}$. Wlog., we assume that $\mathcal{A}$ never asks a query to which it already knows the answer. The goal of $\mathcal{A}$ in this scenario is to determine the result of the coin toss, i.e., to distinguish between the real encryptions with Π and random one.

Relation to Privacy and Integrity Notions. Rogaway and Shrimpton showed in [129] that the CCA3 advantage of an adversary on an AE scheme Π can be upper bounded by the sum of the maximal advantage of an adversary on the integrity of Π, and the maximal advantage of a chosen-plaintext adversary on the privacy of Π. Then CCA3-advantage over all adversaries $\mathcal{A}$ that run in time at most t, ask at most q queries of a total length at most ℓ to the available oracles is given by:

$$\mathbf{Adv}_{\Pi}^{\textsf{CCA3}}(q, t, \ell) \leq \mathbf{Adv}_{\Pi}^{\textsf{IND-CPA}}(q, t, \ell) + \mathbf{Adv}_{\Pi}^{\textsf{INT-CTXT}}(q, t, \ell).$$

4.2 Security Notions for On-Line AE Schemes

Definition 7 (OCCA3-Security). *Suppose $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ is an on-line authenticated encryption scheme, and let $P \xleftarrow{\$} OPerm_n$ be a random on-line permutation. Then, we define the* OCCA3 advantage *of a nonce-ignoring adversary $\mathcal{A}$ as*

$$\mathbf{Adv}_{\Pi}^{OCCA3}\mathcal{A} = \left|\Pr\left[K \leftarrow \mathcal{K} : \mathcal{A}^{\mathcal{E}_{\mathcal{K}}(\cdot,\cdot),\mathcal{D}_{\mathcal{K}}(\cdot,\cdot,\cdot)} \Rightarrow 1\right] - \Pr\left[\mathcal{A}^{\mathcal{O}^{P}(\cdot,\cdot),\perp(\cdot,\cdot,\cdot)} \Rightarrow 1\right]\right|,$$

and

$$\mathbf{Adv}_{\Pi}^{OCCA3}(q, \ell, t) = \max_{\mathcal{A}} \left\{\mathbf{Adv}_{\Pi}^{OCCA3}\mathcal{A}\right\}$$

as the maximum advantage over all nonce-ignoring *OCCA3 adversaries that run in time at most t, ask a total maximum of q queries to the encryption and decryption oracles, and whose total query length is at most ℓ blocks.*

Based on the definition above, an on-line authenticated encryption scheme Π is OCCA3-secure if it provides both OPRP-CPA-security *and* INT-CTXT-security.

Corollary 1 (Bound for the OCCA3 Advantage). *Suppose $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ is an online authenticated encryption scheme. Then, it holds that*

$$\mathbf{Adv}_{\Pi}^{\mathsf{OCCA3}}(\mathcal{A}) \leq \mathbf{Adv}_{\Pi}^{\mathsf{OPRP\text{-}CPA}}(q, \ell, t) + \mathbf{Adv}_{\Pi}^{\mathsf{INT\text{-}CTXT}}(q, \ell, t).$$

We borrow the formal OPRP-CCA-notion from Bellare et al. [22,23], which specifies the maximal advantage of an adversary $\mathcal{A}$ with access to both encryption and decryption oracles to distinguish between the output of a on-line cipher Γ under a randomly chosen key K and that of a random permutation.

Definition 8 (OPRP-CCA-Security). *Let K be a k-bit key, P a random on-line permutation, and $\Gamma : \{0,1\}^k \times (\{0,1\}^n)^* \to (\{0,1\}^n)^*$ an on-line cipher. Then, we define the OPRP-CCA-advantage of an adversary $\mathcal{A}$ by*

$$\mathbf{Adv}_{\Gamma}^{\mathsf{OPRP\text{-}CCA}}(\mathcal{A}) = \left|\Pr\left[\mathcal{A}^{\Gamma_K(\cdot),\Gamma_K^{-1}(\cdot)} \Rightarrow 1\right] - \Pr\left[\mathcal{A}^{P(\cdot),P^{-1}(\cdot)} \Rightarrow 1\right]\right|,$$

where the probabilities are taken over $K \xleftarrow{\$} \mathcal{K}$ and $P \xleftarrow{\$} \mathsf{OPerm}_n$. Further, we define $\mathbf{Adv}_{\Gamma}^{\mathsf{OPRP\text{-}CCA}}(q, \ell, t)$ as the maximum advantage over all OPRP-CCA-adversaries $\mathcal{A}$ that run in time at most t, and make at most q queries of total length ℓ to the available oracles.

Quantitative Security Statements. The CAESAR call demanded quantitative claims of security of each submission for privacy and integrity. For the sake of clarity, we employ two complexities for each notion: query and time complexity. The query complexity q represents the logarithm base-2 of the number of blocks that an adversary has to query in order to violate the claimed security goals with probability of 1/2 or greater. The time complexity t reflects the logarithm base-2 of the number of calls to the underlying primitive function that any adversary has to perform in order to break a security goal with probability of 1/2 or higher, if it has only a few ($\ll q$) plaintext-ciphertext pairs. Tables 4-7 show the security bits of each scheme for the privacy and integrity based on their primitives, respectively.

Provable Security. Provable security is another security measurement which designers assure their security claim by proving them in a theoretical way and based on well-established assumptions, e.g., abstracting their underlying primitive by a random PRF/PRP. In this paper, we also indicate which schemes provide a security proof under well-established assumptions.

4.3 Robustness

An AE scheme is called robust if it provides security in both nonce-misuse and decryption misuse setting, also additional security against more general adversaries [75]. Note that security proofs for AE schemes used to rely on two common assumptions:

1. Adversaries are nonce-respecting. I.e., the adversary will never call the encryption function twice with the same nonce.
2. If authentication fails, the adversary learns no information about the unverified plaintexts, not even partial information.

While both aspects are clear and well-understood in theory, they are hard to guarantee in practice. Thus, security issues have been overlooked or ignored in various cases and security applications have been put at high risk. We consider two robustness notions in the established security definitions: resistance against nonce-ignoring adversaries and against leakage of would-be plaintexts or decryption misuse. Like before, we distinguish between online and off-line schemes.

Security Under Nonce Reuse. In a robust setting for nonce-misuse, the nonce is used more than once with minimal security damage. For a scheme to be robust, there is an ongoing discussion about suitable definition of robust AE.

Rogaway and Shrimpton [127] follow a strict interpretation of (nonce-)misuse-resistant AE (MRAE). According to their notion, an MRAE-secure scheme lets adversaries gain no advantage when a nonce repeats, except for noticing when the same message was encrypted multiple times. Clearly, following this interpretation implies that MRAE-secure schemes can not be online.

In contrast, the notion of nonce-misuse resistance by Fleischmann et al. [61] exclusively targeted online ciphers; the authors considered a nonce repetition as an erroneous usage that resistant schemes should provide as a second line of defense against. Following their definition, an online AE scheme is called secure against nonce-ignoring adversaries if all an adversary can learn from repeating nonces is the longest common prefix of messages. Thus, the privacy protection transformed from PRP-CPA to OPRP-CPA security in this case.

To respect both views, we opt for a two-way strategy: for offline schemes we indicate nonce-misuse resistance iff they provide MRAE (which is equivalent to PRP-CPA and INT-CTXT) security [127]; for online schemes, we indicate nonce-misuse resistance iff they provide OPRP-CPA and INT-CTXT security.

Security Under Release of Unverified Plaintexts (RUP). An unverified plaintext is the message that results from decrypting an unauthentic ciphertext. The security arguments for AE schemes usually require that adversaries never learn anything about such unverified plaintexts. However, for larger data streams or low latency requirements, it may be hard or even impossible to buffer the decryption until the tag is verified.

In this setting, decryption algorithm is allowed to return both $\perp$ and an arbitrary piece of sidelong information for the case of invalid ciphertext. This is covered by the notion of separated AE schemes. The output of decryption algorithm does not matter as long as this information does not help the adversary to decrypt valid messages or forge valid ciphertexts.

Again, (at least) two views exist on this problem. It was first concerned by Abed et al. [3] in their notion of decryption-misuse resistance for online AE schemes. Their notion follows from online chosen-ciphertext security (OPRP-CCA-security), which is the strongest form of non-malleability and decryption-misuse resistance that an *online cipher* can provide, i.e., an adversary that manipulates the i-th block will obtain garbled pseudorandom outputs starting from that block.

Later, in [12], Andreeva et al. formalized and generalized this view. They provided several security definitions meant to capture the requirement that, for an invalid ciphertext and a repeated nonce, decryption algorithm releases only harmless information. They introduced two notions of plaintext awareness (PA1, PA2) for privacy and the INT-RUP notion for integrity. Their definitions reflect that no adversary can gain any advantage by having access to a decryption oracle which always returns a plaintext from any ciphertext input.

As for nonce-misuse case, we opt for a two-way strategy. For offline schemes we indicate decryption-misuse resistance iff they provide PRP-CCA security; for online schemes, we indicate decryption-misuse resistance if they offer OPRP-CCA security.

Definition 9 (INT-RUP Advantage). *Let $\Pi = (\mathcal{E}, \mathcal{D}, \mathcal{V})$ be an authenticated encryption scheme with separated decryption and verification. Then, the INT-RUP advantage of a computationally bounded adversary $\mathcal{A}$ that never queries $\mathcal{E}_{\mathcal{K}} \rightarrow \mathcal{V}_{\mathcal{K}}$,for Π is defined as*

$$\mathbf{Adv}_{\Pi}^{\textsf{INT-RUP}}(\mathcal{A}) := \Pr\left[\mathcal{A}^{\mathcal{E}_{\mathcal{K}}, \mathcal{D}_{\mathcal{K}}, \mathcal{V}_{\mathcal{K}}} \textit{ forges}\right],$$

where the probability is defined over the key $\mathcal{K}$ and random coins of $\mathcal{A}$. Forges means the event of verification oracle that returns $\top$ to the adversary.

5 General Overview of Attacks on Candidates.

In this section, we first give general explanation of broken candidates and their analysis. Then we consider analysis and observation of existing candidates.

5.1 Broken Candidates.

57 candidates were submitted for the first round of the CAESAR competition. At the time of writing this paper, 9 candidates are considered broken and withdrawn from the competition. Candidates are as following:

AES-COBRA. AES-COBRA is an authenticated encryption mode based on AES block cipher with the claim of 64-bit security for both privacy and integrity, and 128-bit for both key recovery and tag guessing attacks. But Nandi [108] showed a forgery attack on n-bit blockcipher with only $\mathcal{O}(n)$ queries and success probability about 1/2 which violate the security claim made by the designers.

Calico. Calico is a family of lightweight authenticated encryption with support of associated data. It is basically based on stream cipher ChaCha-14 and 20, MAC function Siphash-2-4, and hash function BLAKE2. The designer claimed 127 bits of security for the confidentiality of plaintext, and 63 bits of security for the integrity. Christoph Dobraunig et al. [52] showed a forgery and key recovery attacks which requires 2^{64} online queries with the success probability of 1 to recover 128-bit key of the MAC.

CBEAM. CBEAM is algorithm for the authenticated encryption which supports associated data. It uses sponge permutation construction. The designer claimed 127-bit of security for privacy and 63-bit for the privacy but Minaud [101] showed a differential attack on the sponge permutation of CBEAM which can be exploited for a forgery with success probability of 2^{-43} which is contrary to the security claim, 2^{-63}, made by the designers.

FASER. FASER is an authenticated encryption scheme which supports two different versions including 128 and 256-bit. The designers claimed full security for the privacy and 64 and 96-bit of security for the integrity for 128 and 256-bit versions, respectively. Xu et al [152] found a correlation attack on FASER-128 with time complexity of 2^{36} and 2^{12} keystream words. They had also distinguishing attack on FASER-128 and 256-bit versions by only 16 and 64 keystream words. Moreover, Feng et al [58] showed that a real-time key recovery attack is possible on the FASER-128 with only 64 key words to recover all possible keys in real-time in personal computer.

HKC. HKC is a authenticated encryption scheme which is based on a stream cipher. It has built in a MAC routine which provides encrypt then MAC procedures. The designers claimed full security of 256-bit for both privacy and integrity but Saarinen [133] showed that, by taking advantage of linear update function, message forgery attack is trivial and security claim will not hold any more.

Marble. Marble is an authenticated encryption algorithm which supports associated data. The designer claimed full security of 128-bit for both privacy and integrity even for decryption misuse setting, but Fuhr et al [63] showed a simple forgery attack on mode of operation of Marble by using only 2^{64} chosen plaintext queries which violate the security claim made by the designer. They could also recover secret key by using 2^{32} additional decryption queries in the decryption misuse setting. After this attack, the designer modified the mask process but then Lu [98] showed that the modified version is also still vulnerable to both forgery and key-recovery attacks.

McMambo. McMambo is a block-cipher mode of operation based on Mambo cipher. The designer claimed 128-bit of security for the privacy and 64-bit for the integrity. The designer claimed that Mambo block cipher is indistinguishable from the random oracle with a fixed key but Neves [111] showed that there is a iterative differential with probability of 2^{-2} over the full double round of McMambo that can lead to a forgery attack with probability of 2^{-24} which is contrary to the security claim made by the designer.

PAES. PAES is an authenticated encryption algorithm which has two different versions based on the state size namely PAES-4 and PAES-8. The designers claimed 128-bit of security for both privacy and integrity for either version in nonce-respecting model, and only 128-bit for integrity of PAES-8 in nonce-ignoring setting, but Sasaki et al [137] showed a practical universal forgery attack on PAES-8 in nonce-ignoring setting with only 2^{11} encryption queries and computational cost.

PANDA. PANDA is a family of authenticated ciphers which has two versions of PANDA-s and PANDA-b. The designers claimed 128-bit of security for both privacy and integrity in nonce-respecting setting, and 128-bit for PANDA-b in nonce-ignoring setting and only 128-bit of security for privacy of PANDA-s with no security for integrity. Sasaki et al [136] showed a forgery attack in nonce-ignoring setting of PANDA-s with 2^{64} computational cost and negligible memory. Also Feng et al [151] showed another practical forgery and state recovery attack on PANDA-s with time complexity of 2^{41} under known-plaintext-attack with only 137 pairs and negligible memory. Both attacks by Sasaki and Feng violate the security claim made by the designers.

5.2 Second Round Candidates.

July 2015 the committee members of CAESAR announced the second round candidates. From there, 30 out of 48 non-broken candidates could make to the second round. 13 out of 30 candidates are based on block cipher as shown in Table 2, 3 dedicated scheme, 3 stream cipher, 2 permutation, 8 sponge, and one compression function-base, as shown in Table 3.

5.3 Third Round Candidates.

Recently the committee members announced the third round candidates. From there, 15 out of 30 second-round candidates were chosen for the third round. The two candidates AES-COPA and ELmD were merged as COLM for the third round, and SILC and CLOC considered as one candidate.

5.4 External Analysis of Non-broken Candidates.

In this section we summarize all external analysis and observations of candidates that we could find until the time of writing this paper.

Construction	Candidate	External Cryptanalysis	Reference
Block-cipher-based	++AE	Forgery	[142]
	AES-COPA	Universal Forgery	[98,109]
	AES-JAMBU	Distinguish	[116,117]
	AES-CMCC	Distinguish, Forgery	[17,20]
	AEZ	Forgery and Recovery	[64]
	AVALANCHE	Forgery, Key Recovery	[16,39]
	CBA	Distinguish	[54]
	ELmD	Key Recovery on ELmD with 6R-AES	[21]
	Julius-ECB	Forgery	[85]
	iFeed	Forgery, Subkey Reccovery	[146]
	LAC	Differential Forgery	[94]
	POET	Weak Keys, Forgery on POET-G	[1,107]
	iSCREAM	Forgery, Weak Keys, Key Recovery	[93,138]
	Silver	Forgery, Key and Plaintext Recovery	[87]
Stream-cipher-based	ACORN	State Recovery, Key Recovery	[97,46]
	Sablier	Key Recovery	[57]
	Wheesht	Distinguish, Key Recovery, Forgery	[42,110]
	Raviyoyla	Distinguish, Forgery	[35,112]
Sponge-based	ICEPOLE	State Recovery, Forgery	[77,50]
	π-cipher	Tag second-preimage, Forgery, Key Recovery on 2.5 round	[95,96,41]
	PRIMATEs	Forgery, Fault Attack, Key Recovery Cube Attack on PRIMATE-APE	[140,134,102,125]
	NORX	Distinguish, State/Key Recovery on 2 round	[48,18]
Permutation-based	Prøst-OTR	Forgery	[51]

Table 1: External Analysis of Candidates.

6 Overview

Tables 2 and 3 list the properties and parameters of block-cipher- and non-block-cipher-based AE schemes.

Candidate	Mode	Masking	Primitive	Features						Security			2nd-R	3rd-R
				Parallelizable Enc/Dec	*Online*	*Inverse-Free*	*Incremental AD/AE*	*Fixed AD reuse*	*Intermediate Tags*	*Security proof*	*Sec. under Nonce-Reuse*	*Sec. under RUP*		
++AE [120]	ECB	AX	AES	●/●	●	–	–/–	–	–	–	–	–	–	–
AES-CMCC [141]	CBC	–	AES	–/●	–	●	–/–	–	–	●	●	–	–	–
AES-COPA [13]	EME	Doubling	AES	●/●	●	●	●/–	●	–	●	●	–	●	●[4]
AES-CPFB [105]	CTR,PFB	–	AES	●/–	●	●	–/–	–	–	●	–	–	–	–
AES-JAMBU [148]	OFB	–	AES	–/–	●	●	–/–	–	–	–	–	–	●	●
AES-OTR [103]	OTR	Doubling	AES	●/●	●	●	●/–	●	–	●	–	–	●	●
AEZ [74]	OTR	–	AES-4	●/●	–	●	●/–	●	–	●	●	●	●	●
AVALANCHE [9]	ECB	–	AES	●/●	●	●	–/–	–	–	●	–	–	–	–
CBA [76]	ECB	Doubling	AES	●/●	●	●	●/–	●	–	–	–	–	–	–
CLOC [78]	CFB	–	AES*	–/–	●	●	–/–	●	–	●	–	–	●	●
Deoxys$^{\neq}$ [83]	TAE	–	Deoxys-BC,AES	●/●	●	–	–/–	–	–	●	–	–	●	●
Deoxys$^{=}$ [83]	EME	–	Deoxys-BC,AES	●/●	●	–	–/–	–	–	●	●	–	●	●
ELmD [49]	EME	Doubling	AES	●/●	●	–	–/–	–	●	●	●	–	●	●
iFeed[AES] [157]	iFeed	Doubling	AES	●/–	●	●	●/–	●	●	●	–	–	–	–
iSCREAM [67]	TAE	–	iSCREAM, SPN	●/●	●	●	–/–	–	–	–	–	–	–	–
Joltik$^{\neq}$ [84]	TAE	–	Joltik-BC,AES	●/●	●	–	–/–	–	–	●	–	–	●	–
Joltik$^{=}$ [84]	EME	–	Joltik-BC,AES	●/●	●	–	–/–	–	–	●	●	–	●	–
Julius-CTR [19]	CTR	GFM	AES	●/●	–	●	–/–	–	–	●	–	–	–	–
Julius-ECB [19]	ECB	GFM	AES	●/●	–	–	–/–	–	–	●	●	–	–	–
KIASU$^{\neq}$ [82]	TAE	–	KIASU-BC,AES	●/●	●	–	–/–	–	–	●	–	–	–	
KIASU$^{=}$ [82]	EME	–	KIASU-BC,AES	●/●	●	–	–/–	–	–	●	●	–	–	–
LAC [156]	LEX	–	L-Block	●/●	●	–	–/–	–	–	–	–	–	–	–
OCB [91]	TAE	Doubling	AES	●/●	●	–	–/–	–	–	●	–	–	●	●
POET [2]	ECB	AES-4/10	AES	○/○	●	●	●/–	●	●	●	●	●	●	–
SCREAM [67]	TAE	–	SCREAM,SPN	●/●	●	●	–/–	–	–	–	–	–	●	–
SHELL [144]	EME	CTR,Doubling	AES	●/●	●	–	–/–	–	–	●	●	–	●	–
SILC [79]	CFB	–	AES*	–/●	●	●	–/–	–	–	●	–	–	●	●
Silver [115]	TAE	–	MAES	●/●	●	–	–/–	–	–	●	–	–	–	–
YAES [40]	CTR	–	AES	●/●	●	●	●/–	●	–	–	–	–	–	–

Table 2: Block-cipher-based candidates. * = Primary recommendation is AES-based, ● = Provides feature, – = Seems not to provide feature, ○ = Pipelineable, 2nd-R and 3rd-R: second and third round.

[4] The AES-COPA and ElmD are merged as COLM for the third round.

Construction	Candidate	Design	Primitive	Features						Security			2nd-R	3rd-R
				Parallelizable Enc/Dec	*Online*	*Inverse-Free*	*Incremental AD/AE*	*Fixed AD reuse*	*Intermediate Tags*	*Security proof*	*Sec. under Nonce-Reuse*	*Sec. under RUP*		
Dedicated	AES-AEGIS [150]	AES	AES-round	●/–	●	●	–/–	–	–	–	–	–	●	●
	MORUS [149]	LRX	MORUS	–/–	●	●	–/–	–	–	–	–	–	●	●
	Tiaoxin [113]	AES [1]	AES-round	●/●	●	●	–/–	–	–	–	–	–	●	●
Stream-cipher-based	ACORN [147]	LFSR	ACORN	●/●	●	●	–/–	–	–	–	–	–	●	●
	Enchilada [72]	–	ChaCha, Rijndael	●/●	●	●	●/–	●	–	●	–	–	–	–
	HS1-SIV [90]	SIV	ChaCha, Poly1305	–/–	–	●	–/–	–	–	●	●	–	●	–
	Raviyoyla [143]	–	MAGv2	–/–	●	●	–/–	–	–	–	–	–	–	–
	Sablier [155]	LFSR	Sablier	●/●	●	●	●/–	●	–	–	–	–	–	–
	TriviA-ck [43]	–	Trivia-SC	●/●	–	●	–/–	–	●	●	–	–	●	–
	Wheesht [99]	ARX	Wheesht	–/–	●	●	–/–	–	–	–	–	–	–	–
CF-based	OMD [45]	–	SHA-256/512	–/–	●	●	●/–	●	–	●	–	–	●	–
Permutation-based	Minalpher [135]	SPN,XEX	Minalpher-P	●/●	●	–	–/–	–	–	●	●	●	●	–
	PAEQ [37]	PPAE	AESQ	●/●	●	●	●/●	●	–	●	●	–	●	–
	Prøst-COPA [89]	SPN,EME	Prøst	●/●	●	●	●/–	●	–	●	●	–	–	–
	Prøst-OTR [89]	SPN,OTR	Prøst	●/●	●	●	●/–	●	–	●	–	–	–	–
Sponge-based	Artemia [8]	SPN	JHAE	–/–	●	●	–/–	–	–	●	–	–	–	–
	Ascon [53]	SPN,Duplex	Ascon	–/–	●	●	–/–	–	–	●	●	–	●	●
	ICEPOLE [106]	Duplex	Keccak-like	●/●	●	●	–/–	–	●	●	–	–	●	–
	Ketje [34]	Duplex	Keccak-f	–/–	●	●	–/–	–	●	●	–	–	●	●
	Keyak [68]	Duplex	Keccak-f	●/●	●	●	–/–	–	●	●	–	–	●	●
	NORX [15]	LRX,Duplex	n.n.	●/●	●	●	–/–	–	–	●	–	–	●	●
	π-cipher [66]	ARX,Duplex	n.n.	●/●	●	●	–/–	–	–	–	–	–	●	–
	PRIMATEs-GIBBON [10]	SPN,Duplex	PRIMATE	–/–	●	●	–/–	–	–	●	–	–	●	–
	PRIMATEs-HANUMAN [10]	SPN,Duplex	PRIMATE	–/–	●	●	–/–	–	–	●	–	–	●	–
	PRIMATEs-APE [10]	SPN,Duplex	PRIMATE	–/–	●	–	●/–	●	–	●	●	●	●	–
	Prøst-APE [89]	SPN,Duplex	Prøst	–/–	●	–	●/–	●	–	–	●	●	–	–
	STRIBOB [131]	Duplex	Streebog	–/–	●	●	–/–	–	–	●	–	–	●	–

Table 3: Candidates based on dedicated, stream ciphers, compression functions, (non-sponge) permutations, and sponges in particular. n.n. = Unnamed custom primitive, ● = Provides feature, – = Seems not to provide feature.

Candidate	**Parameters**			**Privacy**	**Integrity**
	k	ν	t	q/t	q/t
++AE	128	64	128	64/128	64/126.75
AES-CMCC-32-64	128	32*	64	64/128	64/128
AES-CMCC-32-32	128	32*	32	64/128	32/128
AES-CMCC-16-32	128	16*	32	64/128	16/128
AES-CMCC-32-16	128	32*	16	64/128	32/128
AES-CMCC-16-16	128	16*	16	64/128	16/128
AES-COPA	128	128	128	64/128	64/128
AES-CPFB	128	96	128	64/128	64/128
AES-JAMBU	128	64	64	64/128	64/128
AES-OTR-128	128	96	128	64/128	128/128
AES-OTR-256	256	96	128	64/256	128/256
AEZ	128	96	128	61/128	128/128
AVALANCHE-512	512	160	128	103/256	127/256
AVALANCHE-448	448	128	128	71/192	127/192
AVALANCHE-384	384	80	128	55/128	127/128
CBA-128-32	128	96	32	47/128	47/128
CBA-128-64	128	96	64	63/128	63/128
CBA-128-96	128	96	96	63/128	63/128
CBA-192-64	192	96	64	47/192	47/192
CBA-256-96	256	96	96	63/256	63/256
CLOC-AES-12	128	96	64	64/128	64/128
CLOC-AES-8	128	64	64	64/128	64/128
CLOC-TWINE-6	80	48	32	32/ 80	32/ 80
Deoxys$^{\neq}$-128-128	128	64	128	64/128	128/128
Deoxys$^{\neq}$-256-128	256	64	128	128/256	128/256
Deoxys$^{=}$-128-128	128	64	128	64/128	64/128
Deoxys$^{=}$-256-128	256	64	128	64/256	64/256
ELmD-0-f	128	64	128	62.8/128	62.4/128
ELmD-127-f	128	64	128-255	62.8/128	62.3/128
iFeed[AES]-128-96	128	96	128	64/128	128/128
iFeed[AES]-128-104	128	104	128	64/128	128/128

Table 4: Parameter sets for block-cipher-based candidates. * = 128-bit SNM optional.

Candidate	**Parameters**			**Privacy**	**Integrity**
	k	ν	t	q/t	q/t
Joltik$^{\neq}$-64-64	64	32	64	32/ 64	64/ 64
Joltik$^{\neq}$-80-48	80	24	64	24/ 80	64/ 80
Joltik$^{\neq}$-96-96	96	48	64	48/ 96	64/ 96
Joltik$^{\neq}$-128-64	128	32	64	32/128	64/128
Joltik$^{=}$-64-64	64	32	64	32/ 64	32/ 32
Joltik$^{=}$-80-48	80	24	64	24/ 80	24/ 32
Joltik$^{=}$-96-96	96	48	64	48/ 96	48/ 32
Joltik$^{=}$-128-64	128	32	64	32/128	32/ 32
Julius-ECB-R.	128	96	128	64/128	128/128
Julius-ECB-C.	128	64	128	64/128	64/128
Julius-CTR-R.	128	96	128	64/128	64/128
Julius-CTR-C.	128	64	128	64/128	64/128
KIASU$^{\neq}$	128	32	128	64/128	64/128
KIASU$^{=}$	128	32	128	64/128	64/128
LAC	80	64	64	40/ 80	64/ 80
Marble	128	0	128	128/128	128/128
OCB-128-64	128	128	64	64/128	64/ 64
OCB-128-96	128	128	96	64/128	64/ 96
OCB-128-128	128	128	128	64/128	64/128
OCB-192-64	192	128	64	64/192	64/ 64
OCB-192-96	192	128	96	64/192	64/ 96
OCB-192-128	192	128	128	64/192	64/128
OCB-256-64	256	128	64	64/256	64/ 64
OCB-256-96	256	128	96	64/256	64/ 96
OCB-256-128	256	128	128	64/256	64/128
POET-4	128	128	128	64/128	55/128
POET-10	128	128	128	64/128	64/128
SCREAM	128	96	128	64/128	64/128
SHELL-128-64	128	64	128	55/ 80	55/ 80
SHELL-128-80	128	80	128	55/ 80	55/ 80
SILC/AES-8	128	64	64	64/128	64/128
SILC/AES-12	128	96	64	64/128	64/128
SILC/PRESENT	80	48	32	32/ 80	32/ 80
SILC/LED	80	48	32	32/ 80	32/ 80
Silver	128	128	128	64/128	128/128
YAES	128	127	128	48/ 64	55/128

Table 5: Parameter sets for block-cipher-based candidates. ECB-R. = ECB-Regular, ECB-C. = ECB-Compact, CTR-R. = CTR-Regular, CTR-C. = CTR-Compact.

Candidate	Parameters			Privacy	Integrity
	k	ν	t	q/t	q/t
AES-AEGIS-128	128	128	128	64/128	64/128
AES-AEGIS-256	256	256	128	128/256	128/128
MORUS-640	128	128	128	128/128	128/128
MORUS-1280	256	128	128	256/256	128/256
Tiaoxin	128	128	128	128/128	128/128
ACORN-128	128	128	128	64/128	64/128
Enchilada-128	256	64	128	128/128	128/128
Enchilada-256	256	64	128	128/255	128/255
HS1-SIV-Lo	256	96	64	56/256	56/256
HS1-SIV	256	96	128	112/256	112/256
HS1-SIV-Hi	256	96	256	168/256	168/256
Raviyoyla	256	128	128	128/256	128/256
Sablier	80	80	32	40/ 80	32/128
TriviA-ck	128	64	128	64/128	128/128
Wheesht	512	128^*	256	128/256	128/256
OMD	256	256	32-256	127/256	127/256
Minalpher	256	104	128	64/128	128/256
PAEQ-64	64	64	64	64/ 64	64/ 64
PAEQ-80	80	80	80	80/ 80	80/ 80
PAEQ-128	128	96	128	128/128	128/128
PAEQ-160	160	128	160	160/160	160/160
PAEQ-t-128	128	128	512	128/128	128/128
PAEQ-tnm-128	128	256	512	128/128	128/128
Prøst-COPA-128	128	128	256	64/128	64/128
Prøst-COPA-256	256	256	256	128/256	128/256
Prøst-OTR-128	128	64	128	64/128	64/128
Prøst/OTR-256	256	256	256	128/256	128/256

Table 6: Parameter sets for dedicated, stream-cipher-based, compression-function-based, and permutation-based candidates (from top to bottom). * = 128-bit SNM.

Candidate	Parameters			Privacy	Integrity
	k	ν	t	q/t	q/t
Artemia-128	128	128	128	64/128	64/128
Artemia-256	256	256	256	64/128	128/128
Ascon-128	128	128	128	64/128	64/128
Ascon-96	96	96	96	96/ 96	96/ 96
ICEPOLE-128	128	128^*	128	126/128	128/128
ICEPOLE-128a	128	128	128	126/128	128/128
ICEPOLE-256a	256	96	128	62/128	128/128
Ketje/JR	96	80	96	96/128	96/128
Ketje/SR	128	128	128	128/128	128/128
Keyak	128	128	128	123/128	128/128
NORX/32-4-1	128	64	128	64/128	64/128
NORX/64-4-1	256	128	256	128/256	256/256
NORX/32-6-1	128	64	128	64/128	64/128
NORX/64-6-1	256	128	256	128/256	256/256
NORX/64-4-4	256	128	256	64/256	128/256
π-cipher/16-96	96	32^*	128	48/ 96	96/ 96
π-cipher/16-128	128	32^*	128	64/128	128/128
π-cipher/32-128	128	$128^\dagger$	256	64/128	128/128
π-cipher/32-256	256	$128^\dagger$	256	128/256	256/256
π-cipher/64-128	128	$128^\ddagger$	512	64/128	128/128
π-cipher/64-256	256	$128^\ddagger$	512	128/256	256/512
Pr.-HANUMAN-10	80	80	80	80/ 80	80/ 80
Pr.-HANUMAN-15	120	120	120	120/120	120/120
Pr.-GIBBON-10	80	80	80	80/ 80	80/ 80
Pr.-GIBBON-15	120	120	120	120/120	120/120
Pr.-APE-10	160	80	160	80/ 80	80/ 80
Pr.-APE-15	240	120	240	120/120	120/240
Prøst/APE-128	128	64	128	64/128	64/128
Prøst/APE-256	256	128	256	128/256	128/256
STRIBOB	192	128	128	64/191	127/128

Table 7: Parameter sets for sponge-based candidates. Pr. = PRIMATEs, $^*/^\dagger/^\ddagger$ = 128/256/512-bit SNM.

7 Acknowledgments

The authors would like to thank Bart Mennink for valuable comments and the fruitful discussion during the visit of Farzaneh Abed at ESAT KU Leuven. Furthermore, we thank Elena Andreeva, Eik List, and Jakob Wenzel for their helpful comments.

References

1. Mohamed Ahmed Abdelraheem, Andrey Bogdanov, and Elmar Tischhauser. Weak-Key Analysis of POET. Cryptology ePrint Archive, Report 2014/226, 2014. `http://eprint.iacr.org/`.
2. Farzaneh Abed, Scott Fluhrer, John Foley, Christian Forler, Eik List, Stefan Lucks, David McGrew, and Jakob Wenzel. The POET Family of On-Line Authenticated Encryption Schemes. `http://competitions.cr.yp.to/caesar-submissions.html`, 2014.
3. Farzaneh Abed, Scott R. Fluhrer, Christian Forler, Eik List, Stefan Lucks, David A. McGrew, and Jakob Wenzel. Pipelineable On-Line Encryption. In Carlos Cid and Christian Rechberger, editors, *Fast Software Encryption*, LNCS (to appear). Springer, 2014.
4. Farzaneh Abed, Stefan Kölbl, Martin M. Lauridsen, Christian Rechberger, and Tyge Tiessen. Authenticated Encryption Zoo, 2014. `https://aezoo.compute.dtu.dk/`.
5. Martin Ågren, Martin Hell, Thomas Johansson, and Willi Meier. Grain-128a: A New Version of Grain-128 with Optional Authentication. *IJWMC*, 5(1):48–59, 2011.
6. Dmitry Khovratovich Alex Biryukov. PAEQ: Parallelizable Permutation-based Authenticated Encryption. In *International Security Conference*, volume 17, 12-14 October 2014.
7. Nadhem J. AlFardan and Kenneth G. Paterson. Lucky Thirteen: Breaking the TLS and DTLS Record Protocols. In *2013 IEEE Symposium on Security and Privacy, SP 2013, Berkeley, CA, USA, May 19-22, 2013*, pages 526–540. IEEE Computer Society, 2013.
8. Javad Alizadeh, Mohammad Reza Aref, and Nasour Bagheri. Artemia. `http://competitions.cr.yp.to/caesar-submissions.html`, 2014.
9. Basel Alomair. AVALANCHE. `http://competitions.cr.yp.to/caesar-submissions.html`, 2014.
10. Elena Andreeva, Begül Bilgin, Andrey Bogdanov, Atul Luykx, Florian Mendel, Bart Mennink, Nicky Mouha, Qingju Wang, and Kan Yasuda. PRIMATES. `http://competitions.cr.yp.to/caesar-submissions.html`, 2014.
11. Elena Andreeva, Andrey Bogdanov, Martin M. Lauridsen, Atul Luykx, Bart Mennink, Elmar Tischhauser, and Kan Yasuda. AES-COBRA. `http://competitions.cr.yp.to/caesar-submissions.html`, 2014.
12. Elena Andreeva, Andrey Bogdanov, Atul Luykx, Bart Mennink, Nicky Mouha, and Kan Yasuda. How to Securely Release Unverified Plaintext in Authenticated Encryption. In Palash Sarkar and Tetsu Iwata, editors, *ASIACRYPT (1)*, volume 8873 of *LNCS*, pages 105–125. Springer, 2014.
13. Elena Andreeva, Andrey Bogdanov, Atul Luykx, Bart Mennink, Elmar Tischhauser, and Kan Yasuda. AES-COPA. `http://competitions.cr.yp.to/caesar-submissions.html`, 2014.
14. Frederik Armknecht, Helena Handschuh, Tetsu Iwata, and Bart Preneel. Symmetric Cryptography (Dagstuhl Seminar 14021). *Dagstuhl Reports*, 4(1):1–16, 2014.
15. Jean-Philippe Aumasson, Philipp Jovanovic, and Samuel Neves. NORX. `http://competitions.cr.yp.to/caesar-submissions.html`, 2014.
16. Nasour Bagheri, Javad Alizadeh, and Mohammad Reza Aref. A single query forgery on AVALANCHEv1. Cryptographic Competitions Mailing List, 2014.
17. Nasour Bagheri, Javad Alizadeh, and Mohammad Reza Aref. A distinguishing attack on aes-cmcc v1 by only two queries. Cryptographic Competitions Mailing List, 2014.
18. Nasour Bagheri, Tao Huang, Keting Jia, Florian Mendel, and Yu Sasaki. Cryptanalysis of reduced norx. Cryptology ePrint Archive, Report 2016/436, 2016. `http://eprint.iacr.org/`.
19. Lear Bahack. Julius. `http://competitions.cr.yp.to/caesar-submissions.html`, 2014.
20. Guy Barwell. Forgery on Stateless CMCC. Cryptology ePrint Archive, Report 2014/251, 2014. `http://eprint.iacr.org/`.
21. Asli Bay, Oguzhan Ersoy, and Ferhat Karakoç. Universal forgery and key recovery attacks on elmd authenticated encryption algorithm. Cryptology ePrint Archive, Report 2016/640, 2016. `http://eprint.iacr.org/2016/640`.
22. Mihir Bellare, Alexandra Boldyreva, Lars R. Knudsen, and Chanathip Namprempre. Online Ciphers and the Hash-CBC Construction. In Joe Kilian, editor, *CRYPTO*, volume 2139 of *Lecture Notes in Computer Science*, pages 292–309. Springer, 2001.
23. Mihir Bellare, Alexandra Boldyreva, Lars R. Knudsen, and Chanathip Namprempre. On-line Ciphers and the Hash-CBC Constructions. *Journal of Cryptology*, 25(4):640–679, 2012.
24. Mihir Bellare, Oded Goldreich, and Shafi Goldwasser. Incremental cryptography and application to virus protection. In *Proceedings of the Twenty-seventh Annual ACM Symposium on Theory of Computing*, STOC '95, pages 45–56, New York, NY, USA, 1995. ACM.
25. Mihir Bellare and Chanathip Namprempre. Authenticated Encryption: Relations among Notions and Analysis of the Generic Composition Paradigm. In *ASIACRYPT*, volume 1976 of *Lecture Notes in Computer Science*, pages 531–545. Springer, 2000.
26. Mihir Bellare and Chanathip Namprempre. Authenticated Encryption: Relations among Notions and Analysis of the Generic Composition Paradigm. *Journal of Cryptology*, 21(4):469–491, 2008.

27. Dan J. Bernstein. Failures of secret-key cryptography, March 12 2013. Invited talk at FSE 2013 (20th International Workshop on Fast Software Encryption), Singapore.
28. Dan J. Bernstein. CAESAR Call for Submissions, Final, January 27 2014. `http://competitions.cr.yp.to/caesar-call.html`.
29. Daniel J. Bernstein. SUPERCOP, 2014. `http://bench.cr.yp.to/supercop.html`.
30. G. Bertoni, J. Daemen, M. Peeters, and G. Van Assche. Sponge Functions. Ecrypt Hash Workshop 2007, May 2007.
31. G. Bertoni, J. Daemen, M. Peeters, and G. Van Assche. The Keccak SHA-3 submission. Submission to NIST (Round 3), 2011.
32. Guido Bertoni, Joan Daemen, Michael Peeters, and Gilles Van Assche. Duplexing the Sponge: Single-Pass Authenticated Encryption and Other Applications. In Ali Miri and Serge Vaudenay, editors, *Selected Areas in Cryptography*, volume 7118 of *LNCS*, pages 320–337. Springer, 2011.
33. Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. Permutation-based encryption, authentication and authenticated encryption. In *ECRYPT Directions in Authenticated Ciphers (DIAC) 2012. Stockholm*, 5-6 July 2012.
34. Guido Bertoni, Joan Daemen, Michaël Peeters, and Ronny Van Keer Gilles Van Assche. Ketje. `http://competitions.cr.yp.to/caesar-submissions.html`, 2014.
35. Yuan Yao Bin Zhang and Zhenqing Shi. Some properties of the authentication part of raviyoyla v1, and man-in-the-middle attack. Cryptographic Competitions Mailing List, 2014.
36. Alex Biryukov. Design of a New Stream Cipher-LEX. In *New Stream Cipher Designs - The eSTREAM Finalists*, pages 48–56. 2008.
37. Alex Biryukov and Dmitry Khovratovich. PAEQ. `http://competitions.cr.yp.to/caesar-submissions.html`, 2014.
38. A. Bogdanov, F. Mendel, F. Regazzoni, E. Tischhauser, , and V. Rijmen. ALE: AES-Based Lightweight Authenticated Encryption. In *FSE*, 2013.
39. Andrey Bogdanov, Martin M. Lauridsen, and Elmar Tischhauser. Cryptanalysis of avalanchev1. Cryptographic Competitions Mailing List, 2014. `http://martinlauridsen.info/pub/avalanchev1.pdf`.
40. Antoon Bosselaers and Fre Vercauteren. YAES. `http://competitions.cr.yp.to/caesar-submissions.html`, 2014.
41. Christina Boura, Avik Chakraborti, Gaëtan Leurent, Goutam Paul, Dhiman Saha, Hadi Soleimany, and Valentin Suder. Key recovery attack against 2.5-round pi-cipher. Cryptology ePrint Archive, Report 2016/502, 2016. `http://eprint.iacr.org/`.
42. Anne Canteaut and Gaëtan Leurent. Distinguishing and key-recovery attacks against wheesht. Cryptographic Competitions Mailing List, 2014.
43. Avik Chakraborti and Mridul Nandi. TriviA-ck. `http://competitions.cr.yp.to/caesar-submissions.html`, 2014.
44. Faith Chaza, Cameron McDonald, and Roberto Avanzi. FASER: Authenticated Encryption in a Feedback Shift Register. `http://competitions.cr.yp.to/caesar-submissions.html`, 2014.
45. Simon Cogliani, Diana Ştefania Maimuţ, David Naccache, Rodrigo Portella do Canto, Reza Reyhanitabar, Serge Vaudenay, and Damian Vizár. Offset Merkle-Damgård (OMD), A CAESAR Proposal. `http://competitions.cr.yp.to/caesar-submissions.html`, 2014.
46. Henri Gilbert Colin Chaigneau, Thomas Fuhr. Key recovery attack against ACORN in misuse scenarios, year = 2015, howpublished = Cryptographic Competitions Mailing List,.
47. Joan Daemen and Vincent Rijmen. *The Design of Rijndael: AES - The Advanced Encryption Standard.* Springer, 2002.
48. Sourav Das, Subhamoy Maitra, , and Willi Meier. Higher order differential analysis of norx. Cryptology ePrint Archive, Report 2015/186, 2015. `http://eprint.iacr.org/`.
49. Nilanjan Datta and Mridul Nandi. ELmD. `http://competitions.cr.yp.to/caesar-submissions.html`, 2014.
50. Christoph Dobraunig, Maria Eichlseder, and Florian Mendel. Forgery attacks on round-reduced icepole-128, 2015.
51. Christoph Dobraunig, Maria Eichlseder, and Florian Mendel. Related-Key Forgeries for Proest-OTR. In *FSE*, Lecture Notes in Computer Science. Springer, 2015.
52. Christoph Dobraunig, Maria Eichlseder, Florian Mendel, and Martin Schläffer. Forgery and key recovery attacks on calico. Cryptographic Competitions Mailing List, 2014. `http://ascon.iaik.tugraz.at/files/analysis_calico.pdf`.
53. Christoph Dobraunig, Maria Eichlseder, Florian Mendel, and Martin Schläffer. Ascon: A Family of Authenticated Encryption Algorithms. `http://competitions.cr.yp.to/caesar-submissions.html`, 2014.
54. Alexandre Duc. CPA attack on the CBA mode of operation. Cryptographic Competitions Mailing List, 2014.
55. Thai Duong and Juliano Rizzo. Here Come the $\oplus$ Ninjas, 2011. Manuscript, `http://www.hpcc.ecs.soton.ac.uk/~dan/talks/bullrun/Beast.pdf`.
56. Morris Dworkin. *Special Publication 800-38C: Recommendation for Block Cipher Modes of Operation: The CCM Mode for Authentication and Confidentiality.* National Institute of Standards and Technology, U.S. Department of Commerce, May 2005.
57. Xiutao Feng and Fan Zhang. A practical state recovery attack on the stream cipher sablier v1.
58. Xiutao FENG and Fan ZHANG. A realtime key recovery attack on the authenticated cipher FASER128. Cryptology ePrint Archive, Report 2014/258, 2014. `http://eprint.iacr.org/`.

59. Niels Ferguson and Bruce Schneier. A cryptographic evaluation of IPsec. *Counterpane Internet Security, Inc*, 3031, 2000.
60. Niels Ferguson, Doug Whiting, Bruce Schneier, John Kelsey, Stefan Lucks, and Tadayoshi Kohno. Helix - Fast Encryption and Authentication in a Single Cryptographic Primitive. In *FSE*, pages 330–346. Springer-Verlag, 2003.
61. Ewan Fleischmann, Christian Forler, and Stefan Lucks. McOE: A Family of Almost Foolproof On-Line Authenticated Encryption Schemes. In Anne Canteaut, editor, *FSE*, volume 7549 of *LNCS*, pages 196–215. Springer, 2012.
62. Christian Forler, David McGrew, Stefan Lucks, and Jakob Wenzel. Hash-CFB. In *ECRYPT Directions in Authenticated Ciphers (DIAC) 2012. Stockholm*, 5-6 July 2012.
63. Thomas Fuhr, Gaëtan Leurent, and Valentin Suder. Forgery and Key-Recovery Attacks on CAESAR Candidate Marble. January 2015.
64. Thomas Fuhr, Gaëtan Leurent, and Valentin Suder. Collision attacks against caesar candidates – forgery and key-recovery against aez and marble. Cryptology ePrint Archive, Report 2015/1193, 2015. `http://eprint.iacr.org/`.
65. Virgil D. Gligor and Pompiliu Donescu. Fast Encryption and Authentication: XCBC Encryption and XECB Authentication Modes. In *FSE*, pages 92–108, 2001.
66. Danilo Gligoroski, Hristina Mihajloska, Simona Samardjiska, Håkon Jacobsen, Mohamed El-Hadedy, and Rune Erlend Jensen. π-Cipher. `http://competitions.cr.yp.to/caesar-submissions.html`, 2014.
67. Vincent Grosso, Gaëtan Leurent, François-Xavier Standaert, Kerem Varici, François Durvaux, Lubos Gaspar, and Stéphanie Kerckhof. SCREAM and iSCREAM Side Channel Resistant Authenticated Encryption with Masking. `http://competitions.cr.yp.to/caesar-submissions.html`, 2014.
68. Michaël Peeters Guido Bertoni, Joan Daemen, Gilles Van Assche, and Ronny Van Keer. Keyak. `http://competitions.cr.yp.to/caesar-submissions.html`, 2014.
69. Jian Guo. Marble. `http://competitions.cr.yp.to/caesar-submissions.html`, 2014.
70. Shai Halevi. EME*: Extending EME to Handle Arbitrary-Length Messages with Associated Data. In Anne Canteaut and Kapalee Viswanathan, editors, *Progress in Cryptology - INDOCRYPT 2004, 5th International Conference on Cryptology in India, Chennai, India, December 20-22, 2004, Proceedings*, volume 3348 of *LNCS*, pages 315–327. Springer, 2004.
71. Shai Halevi and Phillip Rogaway. A Parallelizable Enciphering Mode. In Tatsuaki Okamoto, editor, *CT-RSA*, volume 2964 of *Lecture Notes in Computer Science*, pages 292–304. Springer, 2004.
72. Sandy Harris. Enchilada. `http://competitions.cr.yp.to/caesar-submissions.html`, 2014.
73. Matt Henricksen, Shinsaku Kiyomoto, and Jiqiang Lu. The HKC authenticated stream cipher. `http://competitions.cr.yp.to/\discretionary{-}{}{}caesar-submissions.html`, 2014.
74. Viet Tung Hoang, Ted Krovetz, and Phillip Rogaway. AEZ. `http://competitions.cr.yp.to/caesar-submissions.html`, 2014.
75. Viet Tung Hoang, Ted Krovetz, and Phillip Rogaway. Robust authenticated-encryption: Aez and the problem that it solves. Cryptology ePrint Archive, Report 2014/793, 2014. `http://eprint.iacr.org/`.
76. Hossein Hosseini and Shahram Khazaei. CBA Mode. `http://competitions.cr.yp.to/caesar-submissions.html`, 2014.
77. Tao Huan, Ivan Tjuawinata, and Hongjun Wu. Differential-Linear Cryptanalysis of ICEPOLE. In *FSE*, Lecture Notes in Computer Science. Springer, 2015.
78. Tetsu Iwata, Kazuhiko Minematsu, Jian Guo, and Sumio Morioka. CLOC: Compact Low-Overhead CFB. `http://competitions.cr.yp.to/caesar-submissions.html`, 2014.
79. Tetsu Iwata, Kazuhiko Minematsu, Jian Guo, Sumio Morioka, and Eita Kobayashi. SILC: SImple Lightweight CFB. `http://competitions.cr.yp.to/caesar-submissions.html`, 2014.
80. Mohammad Reza Aref Javad Alizadeh and Nasour Bagheri. Jhae: An authenticated encryption mode based on jh. Cryptology ePrint Archive, Report 2014/193, 2014. `http://eprint.iacr.org/`.
81. Matt J.B.Robshaw. Stream Cipher, Technical Report TR-70, Version 2.0, RSA Labratories. Technical report, Intel corporation, 1995.
82. Jérémy Jean, Ivica Nikolić, and Thomas Peyrin. KIASU v1.1, 2014. First-round submission to the CAESAR competition, `http://competitions.cr.yp.to/caesar-submissions.html`.
83. Jérémy Jean, Ivica Nikolić, and Thomas Peyrin. Deoxys v1.3, 2015. Second-round submission to the CAESAR competition.
84. Jérémy Jean, Ivica Nikolić, and Thomas Peyrin. Joltik v1.3, 2015. Second-round submission to the CAESAR competition, `http://competitions.cr.yp.to/caesar-submissions.html`.
85. Tianbin Jiang, Qiushi Wang, and Christophe De Cannière. Comments on Julius-ECB. Cryptographic Competitions Mailing List, 2014.
86. Charanjit S. Jutla. Encryption Modes with Almost Free Message Integrity. In Birgit Pfitzmann, editor, *EUROCRYPT*, volume 2045 of *LNCS*, pages 529–544. Springer, 2001.
87. Lei Wang Jérémy Jean, Yu Sasaki. Analysis of the CAESAR Candidate Silver. In *SAC*, Lecture Notes in Computer Science. Springer, 2015.

88. Jonathan Katz and Moti Yung. Unforgeable Encryption and Chosen Ciphertext Secure Modes of Operation. In *FSE*, pages 284–299, 2000.
89. Elif Bilge Kavun, Martin M. Lauridsen, Gregor Leander, Christian Rechberger, Peter Schwabe, and Tolga Yalçin. Prøst. `http://proest.compute.dtu.dk/`, 2014.
90. Ted Krovetz. HS1-SIV. `http://competitions.cr.yp.to/caesar-submissions.html`, 2014.
91. Ted Krovetz and Phillip Rogaway. OCB. `http://competitions.cr.yp.to/caesar-submissions.html`, 2014.
92. Watson Ladd. MCMAMBO V1: A New Kind Of Latin Dance. `http://competitions.cr.yp.to/caesar-submissions.html`, 2014.
93. Gregor Leander, Brice Minaud, and Sondre Rønjom. A Generic Approach to Invariant Subspace Attacks: Cryptanalysis of Robin, iSCREAM and Zorro. Cryptology ePrint Archive, Report 2015/068, 2015. `http://eprint.iacr.org/`.
94. Gaetan Leurent. Differential Forgery Attack against LAC. https://hal.inria.fr/hal-01017048, 2014.
95. Gaëtan Leurent. Tag second-preimage attack against pi-cipher, 2014. `http://hal.inria.fr/hal-00966794`.
96. Gaëtan Leurent and Thomas Fuhr. Observation on pi-cipher. Cryptographic Competitions Mailing List, 2014.
97. Meicheng Liu and Dongdai Lin. Cryptanalysis of lightweight authenticated cipher acorn. Cryptographic Competitions Mailing List, 2014.
98. Jiqiang Lu. On the security of the copa and marble authenticated encryption algorithms against (almost) universal forgery attack. Cryptology ePrint Archive, Report 2015/079, 2015. `http://eprint.iacr.org/`.
99. Peter Maxwell. wheesht. `http://competitions.cr.yp.to/caesar-submissions.html`, 2014.
100. David McGrew and John Viega. The Galois/Counter Mode of Operation (GCM). *Submission to NIST.* ***`http://csrc.nist.gov/CryptoToolkit/modes/proposedmodes/gcm/gcm-spec.pdf`***, 2004.
101. Brice Minaud. Forging Attack on CBEAM. Cryptographic Competitions Mailing List, 2014.
102. Brice Minaudi. Improved beer-recovery attack against ape, 2014. `https://drive.google.com/file/d/0Bxp3rqwoHZKhQ2s3WlZBZkJ5LUE/edit?pli=1`.
103. Kazuhiko Minematsu. AES-OTR. `http://competitions.cr.yp.to/caesar-submissions.html`, 2014.
104. Kazuhiko Minematsu. Parallelizable Rate-1 Authenticated Encryption from Pseudorandom Functions. In Phong Q. Nguyen and Elisabeth Oswald, editors, *EUROCRYPT*, volume 8441 of *Lecture Notes in Computer Science*, pages 275–292. Springer, 2014.
105. Miguel Montes and Daniel Penazzi. AES-CPFB. `http://competitions.cr.yp.to/caesar-submissions.html`, 2014.
106. Paweł Morawiecki, Kris Gaj, Ekawat Homsirikamol, Krystian Matusiewicz, Josef Pieprzyk, Marcin Rogawski, Marian Srebrny, and Marcin Wójcik. ICEPOLE. `http://competitions.cr.yp.to/caesar-submissions.html`, 2014.
107. Mridul Nandi. Forging attacks on two authenticated encryption schemes COBRA and POET. In *Advances in Cryptology - ASIACRYPT 2014 - 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, R.O.C., December 7-11, 2014. Proceedings, Part I*, pages 126–140, 2014.
108. Mridul Nandi. Forging Attacks on two Authenticated Encryptions COBRA and POET. Cryptology ePrint Archive, Report 2014/363, 2014. `http://eprint.iacr.org/`.
109. Mridul Nandi. Revisiting Security Claims of XLS and COPA. Cryptology ePrint Archive, Report 2015/444, 2015. `http://eprint.iacr.org/2015/444`.
110. Samuel Neves. Forgery attacks against wheesht. Cryptographic Competitions Mailing List, 2014.
111. Samuel Neves. McMambo Iterative Differential. Cryptographic Competitions Mailing List, 2014.
112. Samuel Neves. Raviyoyla stream generation bugs. Cryptographic Competitions Mailing List, 2014.
113. Ivica Nikolić. Tiaoxin-346. `http://competitions.cr.yp.to/caesar-submissions.html`, 2014.
114. US Department of Commerce. DES Modes of Operation. Technical Report FIPS PUB 81, US Department of Commerce / National Bureau of Standards, December 1998.
115. Daniel Penazzi and Miguel Montesg. Silver. `http://competitions.cr.yp.to/caesar-submissions.html`, 2014.
116. Thomas Peyrin, Siang Meng Sim, Lei Wang, and Guoyan Zhang. Cryptanalysis of JAMBU. Cryptology ePrint Archive, Report 2014/931, 2014. `http://eprint.iacr.org/`.
117. Thomas Peyrin, Siang Meng Sim, Lei Wang, and Guoyan Zhang. Cryptanalysis of JAMBU. In *FSE*, Lecture Notes in Computer Science. Springer, 2015.
118. Gordon Procter and Carlos Cid. On Weak Keys and Forgery Attacks against Polynomial-based MAC Schemes. In Shiho Moriai, editor, *FSE*, volume 8424 of *Lecture Notes in Computer Science - Lecture Notes in Computer Science*, pages 287–304. Springer, 2013.
119. Qualys Inc. Ssl pulse – survey of the ssl implementation of the most popular web sites, 2014. `https://www.trustworthyinternet.org/ssl-pulse/`.
120. Francisco Recacha. ++AE. `http://competitions.cr.yp.to/caesar-submissions.html`, 2014.
121. Phillip Rogaway. Authenticated-Encryption with Associated-Data. In *ACM Conference on Computer and Communications Security*, pages 98–107, 2002.
122. Phillip Rogaway. Authenticated-encryption with associated-data. In *Proceedings of the 9th ACM Conference on Computer and Communications Security*, CCS '02, pages 98–107, New York, NY, USA, 2002. ACM.
123. Phillip Rogaway. Efficient Instantiations of Tweakable Blockciphers and Refinements to Modes OCB and PMAC. In *ASIACRYPT*, volume 3329 of *Lecture Notes in Computer Science*, pages 16–31. Springer, 2004.

124. Phillip Rogaway. Nonce-Based Symmetric Encryption. In Bimal K. Roy and Willi Meier, editors, *FSE*, volume 3017 of *LNCS*, pages 348–359. Springer, 2004.
125. Phillip Rogaway. Let's not call it mr, 2014. `http://www.cs.ucdavis.edu/~rogaway/beer.pdf`.
126. Phillip Rogaway, Mihir Bellare, John Black, and Ted Krovetz. OCB: A Block-Cipher Mode of Operation for Efficient Authenticated Encryption. In *ACM Conference on Computer and Communications Security*, pages 196–205, 2001.
127. Phillip Rogaway and Thomas Shrimpton. A Provable-Security Treatment of the Key-Wrap Problem. In Serge Vaudenay, editor, *EUROCRYPT*, volume 4004 of *Lecture Notes in Computer Science*, pages 373–390. Springer, 2006.
128. Phillip Rogaway and Thomas Shrimpton. Deterministic Authenticated-Encryption: A Provable-Security Treatment of the Key-Wrap Problem. Cryptology ePrint Archive, Report 2006/221. (Full Version), 2006. `http://eprint.iacr.org/`.
129. Phillip Rogaway and Thomas Shrimpton. A provable-security treatment of the key-wrap problem. In *Proceedings of the 24th Annual International Conference on The Theory and Applications of Cryptographic Techniques*, EUROCRYPT'06, pages 373–390, Berlin, Heidelberg, 2006. Springer-Verlag.
130. Markku-Juhani O. Saarinen. The CBEAMr1 Authenticated Encryption Algorithm. `http://competitions.cr.yp.to/caesar-submissions.html`, 2014.
131. Markku-Juhani O. Saarinen. The STRIBOBr1 Authenticated Encryption Algorithm. http://competitions.cr.yp.to/-caesar-submissions.html, 2014.
132. Markku-Juhani Olavi Saarinen. Cycling Attacks on GCM, GHASH and Other Polynomial MACs and Hashes. In Anne Canteaut, editor, *FSE*, volume 7549 of *Lecture Notes in Computer Science*, pages 216–225. Springer, 2012.
133. Markku-Juhani Olavi Saarinen. Forging Attack on HKC. Cryptographic Competitions Mailing List, 2014.
134. Dehiman Saha, Sukhendu Kuila, and Depanwita Roy Chuwdhury. Misusing misuse resistance in ape. Cryptographic Competitions Mailing List, 2014.
135. Yu Sasaki, Yosuke Todo, Kazumaro Aoki, Yusuke Naito, Takeshi Sugawara, Yumiko Murakami, Mitsuru Matsui, and Shoichi Hirose. Minalpher. `http://competitions.cr.yp.to/caesar-submissions.html`, 2014.
136. Yu Sasaki and Lei Wang. A Forgery Attack against PANDA-s. Cryptology ePrint Archive, Report 2014/217, 2014. `http://eprint.iacr.org/`.
137. Yu Sasaki and Lei Wang. A Practical Universal Forgery Attack against PAES-8. Cryptology ePrint Archive, Report 2014/218, 2014. `http://eprint.iacr.org/`.
138. Siang Meng Sim and Lei Wang. Practical forgery attacks on scream and iscream, 2014. `http://www1.spms.ntu.edu.sg/~syllab/m/images/b/b3/ForgeryAttackonSCREAM.pdf`.
139. Christopher Taylor. The Calico Family of Authenticated Ciphers. `http://competitions.cr.yp.to/caesar-submissions.html`, 2014.
140. Ivan Tjuawinata and Hongjun Wu. Weakness in the authentication of primates-ape. Cryptographic Competitions Mailing List, 2014.
141. Jonathan Trostle. AES-CMCC. `http://competitions.cr.yp.to/caesar-submissions.html`, 2014.
142. Damian Vizár. Forging Attack on ++AE. Cryptographic Competitions Mailing List, 2014.
143. Rade Vuckovac. Raviyoyla. `http://competitions.cr.yp.to/caesar-submissions.html`, 2014.
144. Lei Wang. SHELL. `http://competitions.cr.yp.to/caesar-submissions.html`, 2014.
145. Doug Whiting, Bruce Schneier, Stefan Lucks, and Frédéric Muller. Fast Encryption and Authentication in a Single Cryptographic Primitive. *ECRYPT Stream Cipher Project Report*, 27(200):5, 2005.
146. Elena Andreeva Bart Preneel Willem Schroé, Bart Mennink. A Forgery and Subkey Recovery on CAESAR candidate iFeed. In *SAC*, Lecture Notes in Computer Science. Springer, 2015.
147. Hongjun Wu. ACORN: A Lightweight Authenticated Cipher. `http://competitions.cr.yp.to/caesar-submissions.html`, 2014.
148. Hongjun Wu and Tao Huang. AES-JAMBU. `http://competitions.cr.yp.to/caesar-submissions.html`, 2014.
149. Hongjun Wu and Tao Huang. The Authenticated Cipher MORUS. `http://competitions.cr.yp.to/caesar-submissions.html`, 2014.
150. Hongjun Wu and Bart Preneel. AEGIS: A Fast Authenticated Encryption Algorithm. `http://competitions.cr.yp.to/caesar-submissions.html`, 2014.
151. Fan ZHANG Xiutao FENG and Hui WANG. A practical forgery and state recovery attack on the authenticated cipher PANDA-s. Cryptology ePrint Archive, Report 2014/325, 2014. `http://eprint.iacr.org/`.
152. Chao Xu, Bin Zhang, and Willi Meier. Cryptanalysis of FASER Stream Cipher. Cryptographic Competitions Mailing List, 2014.
153. Dingfeng Ye, Peng Wang, Lei Hu, Liping Wang, Yonghong Xie, Siwei Sun, and Ping Wang. PAES: Parallelizable Authenticated Encryption Schemes based on AES Round Function. `http://competitions.cr.yp.to/caesar-submissions.html`, 2014.
154. Dingfeng Ye, Peng Wang, Lei Hu, Liping Wang, Yonghong Xie, Siwei Sun, and Ping Wang. PANDA. `http://competitions.cr.yp.to/caesar-submissions.html`, 2014.
155. Bin Zhang, Zhenqing Shi, Chao Xu, Yuan Yao, and Zhenqi Li. Sablier. `http://competitions.cr.yp.to/caesar-submissions.html`, 2014.

156. Lei Zhang, Wenling Wu, Yanfeng Wang, Shengbao Wu, and Jian Zhang. Lac: A lightweight authenticated encryption cipher. `http://competitions.cr.yp.to/caesar-submissions.html`, 2014.
157. Liting Zhang, Wenling Wu, Han Sui, and Peng Wang. iFeed[AES]. `http://competitions.cr.yp.to/caesar-submissions.html`, 2014.
158. Yuliang Zheng, Tsutomu Matsumoto, and Hideki Imai. On the Construction of Block Ciphers Provably Secure and Not Relying on Any Unproved Hypotheses. In Gilles Brassard, editor, *Advances in Cryptology - CRYPTO '89, 9th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 1989, Proceedings*, volume 435 of *Lecture Notes in Computer Science*, pages 461–480. Springer, 1989.