Stream:	Internet Research Task Force (IRTF)
RFC:	9771
Category:	Informational
Published:	May 2025
ISSN:	2070-1721
Author:	A. Bozhko, Ed.
	CryptoPro

RFC 9771
Properties of Authenticated Encryption with Associated Data (AEAD) Algorithms

Abstract

Authenticated Encryption with Associated Data (AEAD) algorithms provide both confidentiality and integrity of data. The widespread use of AEAD algorithms in various applications has led to an increased demand for AEAD algorithms with additional properties, driving research in the field. This document provides definitions for the most common of those properties and aims to improve consistency in the terminology used in documentation. This document is a product of the Crypto Forum Research Group.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

An Authenticated Encryption with Associated Data (AEAD) algorithm provides confidentiality for the plaintext to be encrypted and integrity for the plaintext and some associated data (sometimes called "Header"). AEAD algorithms play a crucial role in various applications and have emerged as a significant focus in cryptographic research.

1.1. Background

AEAD algorithms are formally defined in [RFC5116]. The main benefit of AEAD algorithms is that they simultaneously provide data confidentiality and integrity and have a simple unified interface. In contrast to generic compositions of Message Authentication Code (MAC) and encryption algorithms, an AEAD algorithm allows for a reduction in key and state sizes, improving the data processing speed. Most AEAD algorithms come with security analysis, usage guidelines, and reference implementations. Consequently, their integration into high-level schemes and protocols is highly transparent. For instance, AEAD algorithms are mandatory in TLS 1.3 [RFC8446], IPsec Encapsulating Security Payload (ESP) [RFC4303] [RFC8221], and QUIC [RFC9000].

While confidentiality and data integrity (the conventional properties of AEAD algorithms) suffice for many applications, some environments demand other uncommon cryptographic properties. These often require additional analysis and research. As the number of such properties and corresponding research papers grows, inevitable misunderstandings and confusion arise. This is a common situation when related but formally different properties are named identically or when some security properties only have folklore understanding and are not formally defined. Consequently, the risk of misusing AEAD algorithms increases, potentially resulting in security issues.

1.2. Scope

In Section 4 of this document, we provide a list of the most common additional properties of AEAD algorithms. The properties are divided into two categories, namely, security properties (see Section 4.3) and implementation properties (see Section 4.4). We provide a high-level definition for each property. For security properties, we also reference an informative source where a formal game-based security notion is defined; we do not consider security properties for which no game-based formalization exists. When possible, we offer additional information: synonymous names, examples of algorithms that provide the property, applications that might necessitate the property from an AEAD algorithm, references for further reading, and additional notes containing information outside these categories.

The objective of this document is to enhance clarity and establish a common language in the field. In particular, the primary application of the document lies in the following two use cases within the document development process in the IRTF and IETF:

- For an RFC or I-D that defines an AEAD algorithm, it is recommended to use the notations in Section 4 when listing additional properties of the algorithm.
- For an RFC or I-D that defines a generic protocol based on an AEAD algorithm, it is recommended to use the notations in Section 4 if any additional properties are required from the algorithm.

This document represents the consensus of the Crypto Forum Research Group (CFRG). This document is not an IETF product and is not a standard.

2. Conventions Used in This Document

The key words "**MUST**", "**MUST NOT**", "**REQUIRED**", "**SHALL**", "**SHALL NOT**", "**SHOULD**", "**SHOULD NOT**", "**RECOMMENDED**", "**NOT RECOMMENDED**", "**MAY**", and "**OPTIONAL**" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

3. AEAD Algorithms

This section gives a conventional definition of an AEAD algorithm following [RFC5116].

Definition:

An AEAD algorithm is defined by two operations, which are authenticated encryption and authenticated decryption:

- A deterministic operation of authenticated encryption has four inputs, each a binary string: a secret key K of a fixed bit length, a nonce N, associated data A, and a plaintext P. The plaintext contains the data to be encrypted and authenticated, and the associated data contains the data to be authenticated only. Each nonce value **MUST** be unique in every distinct invocation of the operation for any particular value of the key. The authenticated encryption operation outputs a ciphertext C.
- A deterministic operation of authenticated decryption has four inputs, each a binary string: a secret key K of a fixed bit length, a nonce N, associated data A, and a ciphertext C. The operation verifies the integrity of the ciphertext and associated data and decrypts the ciphertext. It returns a special symbol FAIL if the inputs are not authentic; otherwise, the operation returns a plaintext P.

We note that specifications of AEAD algorithms that use authentication tags to ensure integrity may define the authentication tag as an independent output of the encryption operation and an independent input of the decryption operation. Throughout this document, by default, we consider the authentication tag as part of the ciphertext.

For more details on the AEAD definition, please refer to [RFC5116].

Throughout this document, by default, we consider nonce-based AEAD algorithms, which have an interface as defined above, and we give no other restrictions on their structure. However, some properties considered in the document apply only to particular classes of such algorithms, like AEAD algorithms based on block ciphers (such algorithms use a block cipher as a building block). If that is the case, we explicitly point that out in the corresponding section.

4. AEAD Properties

4.1. Classification of Additional AEAD Properties

In this document, we employ a high-level classification of additional properties. This classification aims to provide insight into how one can benefit from each property. The additional properties are categorized into one of these two groups:

- Security properties: We classify a property as a security property if it either takes into account new threats or extends adversarial capabilities, in addition to those posed by the typical nonce-respecting adversary whose goal is to compromise confidentiality or data integrity.
- Implementation properties: We classify a property as an implementation property if it enables more efficient implementations of the AEAD algorithm in specific cases or environments.

We note that some additional properties of AEAD algorithms found in the literature could not be allocated to either of these two groups. The observation is that such properties require an extension of the conventional AEAD interface. We refer to these properties as "additional functionality properties" and define the corresponding group as follows:

- Additional functionality properties: We classify a property as an additional functionality property if it introduces new features in addition to the standard AEAD.

With the extension of the conventional AEAD interface, each additional functionality property defines a new class of cryptographic algorithms. Consequently, the basic threats and adversarial capabilities must be redefined for each class. As a result, additional functionality properties consider the basic threats and adversarial capabilities for their class of algorithms, in contrast to security properties, which consider the extended ones. For this reason, we do not focus on additional functionality properties in this document. However, for the sake of completeness, in Appendix A, we briefly present two classes of AEAD algorithms with additional functionality.

4.2. Conventional Properties

In this section, we recall the conventional properties of an AEAD algorithm. Active nonce-respecting adversaries in a single-key setting are considered.

We say that an AEAD algorithm provides security if it provides the conventional properties listed in this section.

4.2.1. Confidentiality

Definition:
  An AEAD algorithm guarantees that the plaintext is not available to an active, nonce-respecting adversary.

Security notions:
  IND-CCA [BN08] (or IND-CCA2 [S04])

Synonyms:
  Message privacy

Notes:
  Confidentiality against passive adversaries can also be considered. The corresponding security notion is IND-CPA [BN08] [R02].

Further reading:
  [R02], [BN08], [S04]

4.2.2. Data Integrity

Definition:
  An AEAD algorithm allows one to ensure that the ciphertext and the associated data have not been changed or forged by an active, nonce-respecting adversary.

Security notions:
INT-CTXT [BN08] (or AUTH [R02])

Synonyms:
Message authentication, authenticity

Further reading:
[R02], [BN08], [S04]

4.2.3. Authenticated Encryption Security

Definition:
An AEAD algorithm provides confidentiality and data integrity against active, nonce-respecting adversaries.

Security notions:
IND-CPA and INT-CTXT [BN08] [R02] (or equivalently, IND-CCA3 [S04])

Notes:
Please refer to [AEAD-LIMITS] for usage limits on modern AEAD algorithms used in IETF protocols.

Further reading:
[R02], [BN08], [S04]

4.3. Security Properties

4.3.1. Blockwise Security

Definition:
An AEAD algorithm provides security even if an adversary can adaptively choose the next part of the plaintext depending on already-computed ciphertext parts during an encryption operation.

Security notions:
D-LORS-BCPA for confidentiality against passive adversaries, B-INT-CTXT for integrity [EV17]; OAE1 [HRRV15] (a stronger notion; originally OAE (Online Authenticated Encryption) in [FFL12])

Examples:
Deoxys [JNPS21], SAEF [ABV21]

Notes:
Blockwise security is highly relevant for streamable AEAD algorithms (see Section 4.4.8). The OAE1 security notion [HRRV15] and the OAE2 notion [HRRV15] are tailored for streamable AEAD algorithms. OAE1 was first defined in [FFL12] under the name OAE; however, it contained a glitch, and the reformulated definition was presented in [HRRV15]. Blockwise security follows from security in the OAE notion [EV17]. For a discussion on security notions for streamable AEAD algorithms, see [HRRV15].

Applications:
: Real-time streaming protocols, encryption on resource-constrained devices

Further reading:
: [EV17], [JMV2002], [FJMV2004], [HRRV15]

4.3.2. Full Commitment

Definition:
: An AEAD algorithm guarantees that it is hard to find two or more different tuples of the key, nonce, associated data, and plaintext such that they encrypt to the same ciphertext. In other words, an AEAD scheme guarantees that a ciphertext is a commitment to all inputs of an authenticated encryption operation.

Security notions:
: CMT-4 [BH22], generalized CMT for a restricted setting (see the notes below) [MLGR23]

Examples:
: Ascon [DEMS21a] [DEMS21b] [YSS23], full committing versions of Galois/Counter Mode (GCM) and GCM-SIV [BH22], generic constructions [BH22] and [CR22]

Notes:
: Full commitment can be considered in a weaker setting, where certain restrictions on the tuples produced by an adversary are imposed [MLGR23]. For instance, an adversary must find tuples that all share the same associated data value. In such cases, an AEAD algorithm is said to provide full commitment in a restricted setting. The imposed restrictions **MUST** be listed.

Applications:
: Message franking [GLR17]

Further reading:
: [BH22], [CR22], [MLGR23]

4.3.3. Key Commitment

Definition:
: An AEAD algorithm guarantees that it is hard to find two or more different keys and the same number of potentially equal triples of nonce, associated data, and plaintext such that they encrypt to the same ciphertext under corresponding keys. In other words, an AEAD scheme guarantees that a ciphertext is a commitment to the key used for an authenticated encryption operation.

Security notions:
: CMT-1 [BH22]

Synonyms:
: Key robustness, key collision resistance

Examples:
: Ascon [DEMS21a] [DEMS21b] [YSS23], generic constructions from [BH22] and [CR22]

Notes:
: Key commitment follows from full commitment. Full commitment does not follow from key commitment [BH22].

Applications:
: Password-Authenticated Key Exchange, password-based encryption [LGR21], key rotation, envelope encryption [ADGKLS22]

Further reading:
: [BH22], [CR22], [FOR17], [LGR21], [GLR17]

4.3.4. Leakage Resistance

Definition:
: An AEAD algorithm provides security even if some additional information about computations of an encryption (and possibly decryption) operation is obtained via side-channel leakages.

Security notions:
: CIL1 [GPPS19] (CIML2 [BPPS17] with leakages in decryption) for integrity, CCAL1 [GPPS19] (CCAmL2 [GPPS19] with leakages in decryption) for authenticated encryption security

Examples:
: Ascon [DEMS21a] [DEMS21b] (security under CIML2 and CCAL1 notions [B20]), TEDT [GPPS19]

Notes:
: Leakages during AEAD operation executions are implementation-dependent. It is possible to implement symmetric algorithms in a way that every possible physical leakage is entirely independent of the secret inputs of the algorithm (for example, with a masking technique [CJRR99]), meaning the adversary doesn't gain any additional information about the algorithm's computation via side-channel leakages. We say that an AEAD algorithm doesn't provide leakage resistance if it can only achieve leakage resistance with such an implementation. Leakage-resistant AEAD algorithms aim to place requirements on implementations that are as mild as possible to achieve leakage resistance. These requirements **SHOULD** be listed.

  Confidentiality of plaintext in the presence of leakages in the encryption operation is unachievable if an adversary can repeat the nonce used to encrypt the plaintext in other encryption queries. Confidentiality can be achieved only for plaintexts encrypted with fresh nonces (analogously to nonce-misuse resilience; see Section 4.3.7). For further discussions, see [GPPS19] and [B20].

  For primitive-based AEAD algorithms, key evolution (internal re-keying [RFC8645]) can contribute to achieving leakage resistance with leakages in encryption. Confidentiality in the presence of decryption leakages can be achieved by two-pass AEAD algorithms with key evolution, which compute independent ephemeral key values for encryption and tag generation, where the computation of these keys is implemented without any leakages. For more discussion on achieving leakage resistance, see [B20].

Leakage Resilience, a well-known weaker property introduced in [BMOS17], can also be considered. However, following the framework established in [GPPS19] and [B20], this document makes a conscious choice to focus on the stronger Leakage Resistance for its enhanced practicality and comprehensiveness.

Applications:
Encryption on smart cards, Internet-of-Things devices, or other constrained devices

Further reading:
[GPPS19], [B20], [BPPS17], [BMOS17]

4.3.5. Multi-user Security

Definition:
The security of an AEAD algorithm degrades slower than linearly with an increase in the number of users.

Security notions:
mu-ind [BT16]

Examples:
AES-GCM [D07], ChaCha20-Poly1305 [RFC8439], AES-GCM-SIV [RFC8452], AEGIS [AEGIS-AEAD]

Notes:
For any AEAD algorithm, security degrades no worse than linearly with an increase in the number of users [BT16]. However, for some applications with a significant number of users, better multi-user guarantees are required. For example, in the TLS 1.3 protocol, AEAD algorithms are used with a randomized nonce (deterministically derived from a traffic secret and a sequence number) to address this issue. Using nonce randomization in block cipher counter-based AEAD modes can contribute to multi-user security [BT16]. Multi-user usage limits for AES-GCM and ChaCha20-Poly1305 are provided in [AEAD-LIMITS].

A weaker security notion, multi-user key recovery, is also introduced and thoroughly studied in [BT16]. While this document focuses on indistinguishability for security notions, key recovery might be relevant and valuable to study alongside indistinguishability.

Applications:
Data transmission layer of secure communication protocols (e.g., TLS, IPsec, the Secure Real-time Transport Protocol (SRTP), etc.)

Further reading:
[BT16], [HTT18], [LMP17], [DGGP21], [BHT18]

4.3.6. Nonce Hiding

Definition:
: An AEAD algorithm provides confidentiality for the nonce value used to encrypt plaintext. The algorithm includes information about the nonce in the ciphertext and doesn't require the nonce as input for the decryption operation.

Security notions:
: AE2 [BNT19]

Examples:
: Hide-Nonce (HN) transforms [BNT19]

Notes:
: As discussed in [BNT19], adversary-visible nonces might compromise message and user privacy, similar to the way any metadata might. As pointed out in [B13], even using a counter as a nonce value might compromise privacy. Designing a privacy-preserving way to manage nonces might be a challenging problem for an application.

Applications:
: Any application that can't rely on a secure "out-of-band" nonce communication

Further reading:
: [BNT19]

4.3.7. Nonce Misuse

Definition:
: An AEAD algorithm provides security (resilience or resistance) even if an adversary can repeat nonces in its encryption queries. Nonce misuse resilience and resistance are defined as follows:

  Nonce misuse resilience: Security is provided for messages encrypted with non-repeated (fresh) nonces (correctly encrypted messages).

  Security notions:
  : Chosen-Plaintext Attack (CPA) resilience (confidentiality), authenticity resilience (integrity), Chosen-Ciphertext Attack (CCA) resilience (authenticated encryption) [ADL17]

  Examples:
  : ChaCha20-Poly1305 [RFC8439], AES-GCM [D07] (only confidentiality)

  Nonce misuse resistance: Security is provided for all messages that were not encrypted with the same nonce value more than once.

  Security notions:
  : MRAE [RS06]

  Examples:
  : AES-GCM-SIV [RFC8452], Deoxys-II [JNPS21]

Notes:
Synthetic Initialization Vector (SIV) construction [RS06] is a generic construction that provides nonce misuse resistance.

Notes:
Nonce misuse resilience follows from nonce misuse resistance. Nonce misuse resistance does not follow from nonce misuse resilience.

Applications:
Any application where nonce uniqueness can't be guaranteed, security against fault-injection attacks and malfunctions, processes parallelization, full disk encryption

Further reading:
[RS06], [ADL17], [IIM25]

4.3.8. Quantum Security

Definition:
An AEAD algorithm provides security (in a Q1 or Q2 model) against a quantum adversary. Q1 and Q2 models are defined as follows:

Q1 model: An adversary has access to local quantum computational power. It has classical access to encryption and decryption oracles.

Synonyms:
Post-quantum security

Examples:
AES-GCM [D07], ChaCha20-Poly1305 [RFC8439], OCB [RFC7253], Multilinear Galois Mode (MGM) [RFC9058], AES-GCM-SIV [RFC8452], AEGIS [AEGIS-AEAD]

Q2 model: An adversary has access to local quantum computational power. It has quantum access to encryption and decryption oracles, i.e., it can query encryption and decryption oracles with quantum superpositions of inputs to receive quantum superpositions of the outputs.

Synonyms:
Superposition-based quantum security

Examples:
QCB [BBCLNSS21]

Notes:
Most symmetric cryptographic algorithms that are secure in the classical model provide quantum security in the Q1 model, i.e., they are post-quantum secure. Security in the Q1 setting corresponds to security against "harvest now, decrypt later" attacks. Security in Q1 follows from security in Q2; the converse does not hold. For discussions on the relevance of the Q2 model, please see [G17].

Further reading:
[KLLNP16], [BBCLNSS21], [G17]

4.3.9. Reforgeability Resilience

Definition:
An AEAD algorithm guarantees that once a successful forgery for the algorithm has been found, it is still hard to find any subsequent forgery.

Security notions:
j-Int-CTXT [FLLW17]

Examples:
Deoxys [JNPS21], AEGIS [AEGIS-AEAD], Ascon [DEMS21a] [DEMS21b]

Applications:
Voice over IP (VoIP), real-time streaming in a lightweight setting, applications that require small ciphertext expansion (i.e., short tags)

Further reading:
[BC09], [FLLW17]

4.3.10. Release of Unverified Plaintext (RUP) Integrity

Definition:
An AEAD algorithm provides data integrity even if plaintext is released for every ciphertext, including those with failed integrity verification.

Security notions:
INT-RUP [A14]

Examples:
GCM [IIM25], GCM-RUP [ADL17]

Applications:
Decryption with limited memory [FJMV2004], real-time streaming protocols

Notes:
In [ADL17], a generic approach to achieve INT-RUP security is introduced.

In the provided definition, we only consider integrity in the RUP setting, since confidentiality, in the usual sense, is unachievable under RUP. In [A14], the notion of "Plaintext Awareness" is introduced, capturing the best possible confidentiality under RUP in the following sense: "the adversary cannot gain any additional knowledge about the plaintext from decryption queries besides what it can derive from encryption queries".

Further reading:
[A14], [ADL17], [IIM25]

4.4. Implementation Properties

4.4.1. Hardware Efficient

Definition:
: An AEAD algorithm ensures optimal performance when operating on hardware that complies with the specified requirements.

Notes:
: Various classes of hardware may be taken into consideration. Certain algorithms are tailored to minimize the area of dedicated hardware implementations, while others are intended to capitalize on general-purpose CPUs, with or without specific instruction sets. It is **RECOMMENDED** to specify the minimum platform requirements for the AEAD to fulfill its intended purpose, as well as to match its performance and security claims.

4.4.2. Inverse-Free

Definition:
: An AEAD algorithm based on a given primitive can be implemented without invoking the inverse of that primitive.

Examples:
: AES-GCM [D07], ChaCha20-Poly1305 [RFC8439], MGM [RFC9058], AEGIS [AEGIS-AEAD]

Notes:
: In a sponge-based AEAD algorithm, an underlying permutation is viewed as a primitive.

4.4.3. Lightweight

Definition:
: An AEAD algorithm can be efficiently and securely implemented on resource-constrained devices. In particular, it meets the criteria required in the NIST Lightweight Cryptography competition [MBTM17].

Examples:
: OCB [RFC7253], Ascon [DEMS21a] [DEMS21b]

Further reading:
: [MBTM17]

4.4.4. Parallelizable

Definition:
: An AEAD algorithm can fully exploit the parallel computation infrastructure. In other words, a parallelizable AEAD algorithm allows for the computation of ciphertext segments (plaintext segments for decryption) in parallel, meaning that ciphertext segments are computed independently.

Synonyms:
Pipelineable

Examples:
AES-GCM [D07], ChaCha20-Poly1305 [RFC8439], OCB [RFC7253], MGM [RFC9058], AEGIS [AEGIS-AEAD]

Further reading:
[C20]

4.4.5. Setup-Free

Definition:
An AEAD algorithm's operations can be implemented in a way that using a new key incurs either no overhead or negligible overhead compared to the reuse of a previous key. Overhead may involve additional computations or increased storage space, such as precomputing a key schedule for a block cipher.

Examples:
ChaCha20-Poly1305 [RFC8439], AEGIS [AEGIS-AEAD], Ascon [DEMS21a] [DEMS21b]

4.4.6. Single Pass

Definition:
An AEAD algorithm encryption (decryption) operation can be implemented with a single pass over the plaintext (ciphertext).

Examples:
AES-GCM [D07], ChaCha20-Poly1305 [RFC8439], OCB [RFC7253], MGM [RFC9058], AEGIS [AEGIS-AEAD]

4.4.7. Static Associated Data Efficient

Definition:
An AEAD algorithm allows precomputation for static (or repeating) associated data so that static associated data doesn't significantly contribute to the computational cost of encryption.

Examples:
AES-GCM [D07], ChaCha20-Poly1305 [RFC8439], OCB [RFC7253]

4.4.8. Streamable

Definition:
An AEAD algorithm encryption (decryption) operation can be implemented with constant memory usage and a single one-direction pass over the plaintext (ciphertext), writing out the result during that pass.

Synonyms:
: Online

Examples:
: AES-GCM [D07], ChaCha20-Poly1305 [RFC8439], OCB [RFC7253], MGM [RFC9058], AEGIS [AEGIS-AEAD], Ascon [DEMS21a] [DEMS21b]

Applications:
: Real-time streaming protocols, resource-constrained devices

Notes:
: Blockwise security (see Section 4.3.1) and RUP integrity (see Section 4.3.10) might be relevant security properties for streamable AEAD algorithms in certain applications.

Further reading:
: [HRRV15], [FJMV2004]

5. Security Considerations

This document gives high-level definitions of AEAD properties. For each security property, we provide an informational reference to a game-based security notion (or security notions if there are separate notions for integrity and confidentiality) that formalizes the property. We only consider game-based notions and security properties that can be formalized using this approach. However, there are different approaches to formalizing AEAD security, like the indifferentiability framework [BM18]; security in such notions should be studied separately.

For some properties, examples of AEAD algorithms that provide them are given, with standardized AEAD algorithms preferred for commonly encountered properties. However, for certain properties, only non-standardized algorithms exist. Implementing such algorithms requires careful consideration, and it is advised to contact the algorithm designers for reference implementations and implementation guidelines.

Every claimed security property of an AEAD algorithm **MUST** undergo security analysis within a relevant notion. It's **RECOMMENDED** to use the security notions referenced in the document. If an alternative notion is used, proof of equivalence **MUST** exist, or use of a non-equivalent notion **SHOULD** be indicated. For security properties that extend adversarial capabilities, consideration of integrity and confidentiality separately may be relevant. If the algorithm provides only one of these, that **SHOULD** be indicated.

When specifying security requirements for an AEAD algorithm in an application, it **SHOULD** be indicated, for every required security property, whether only integrity or confidentiality is necessary. Additionally, for each security property, it **SHOULD** be specified whether an analysis in an alternative security notion is required. We also note that some additional properties come with trade-offs in terms of classical security and efficiency, and they may only be supported in non-standardized or modified AEAD algorithms. This immediately implies challenges in deployment and interoperability. In an application, the requirements for additional AEAD properties **SHOULD** be highly motivated and justified, and all trade-offs should be carefully considered.

6. IANA Considerations

This document has no IANA actions.

7. References

7.1. Normative References

[D07] Dworkin, M., "Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC", NIST SP 800-38D, DOI 10.6028/NIST.SP.800-38D, 2007, <https://csrc.nist.gov/pubs/sp/800/38/d/final>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC5116] McGrew, D., "An Interface and Algorithms for Authenticated Encryption", RFC 5116, DOI 10.17487/RFC5116, January 2008, <https://www.rfc-editor.org/info/rfc5116>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

7.2. Informative References

[A14] Andreeva, E., Bogdanov, A., Luykx, A., Mennink, B., Mouha, N., and K. Yasuda, "How to Securely Release Unverified Plaintext in Authenticated Encryption", Advances in Cryptology - ASIACRYPT 2014, Lecture Notes in Computer Science, vol. 8873, pp. 105-125, DOI 10.1007/978-3-662-45611-8_6, 2014, <https://doi.org/10.1007/978-3-662-45611-8_6>.

[ABV21] Andreeva, E., Bhati, A.S., and D. Vizár, "Nonce-Misuse Security of the SAEF Authenticated Encryption Mode", Selected Areas in Cryptography (SAC 2020), Lecture Notes in Computer Science, vol. 12804, pp. 512-534, DOI 10.1007/978-3-030-81652-0_20, 2021, <https://doi.org/10.1007/978-3-030-81652-0_20>.

[ADGKLS22] Albertini, A., Duong, T., Gueron, S., Kölbl, S., Luykx, A., and S. Schmieg, "How to Abuse and Fix Authenticated Encryption Without Key Commitment", 31st USENIX Security Symposium (USENIX Security 22), pp. 3291-3308, 2022.

[ADL17] Ashur, T., Dunkelman, O., and A. Luykx, "Boosting Authenticated Encryption Robustness with Minimal Modifications", Advances in Cryptology - CRYPTO 2017, Lecture Notes in Computer Science, vol. 10403, pp. 3-33, DOI 10.1007/978-3-319-63697-9_1, 2017, <https://doi.org/10.1007/978-3-319-63697-9_1>.

[AEAD-LIMITS] Günther, F., Thomson, M., and C. A. Wood, "Usage Limits on AEAD Algorithms", Work in Progress, Internet-Draft, draft-irtf-cfrg-aead-limits-10, 8 April 2025, <https://datatracker.ietf.org/doc/html/draft-irtf-cfrg-aead-limits-10>.

[AEGIS-AEAD] Denis, F. and S. Lucas, "The AEGIS Family of Authenticated Encryption Algorithms", Work in Progress, Internet-Draft, draft-irtf-cfrg-aegis-aead-16, 17 February 2025, <https://datatracker.ietf.org/doc/html/draft-irtf-cfrg-aegis-aead-16>.

[B13] Bernstein, D. J., "Re: wondering the secret message number", Message to the Cryptographic Competitions Google Group, 2013, <https://groups.google.com/d/msg/crypto-competitions/n5ECGwYr6Vk/bsEfPWqSAU4J>.

[B20] Bellizia, D., Bronchain, O., Cassiers, G., Grosso, V., Guo, C., Momin, C., Pereira, O., Peters, T., and F.-X. Standaert, "Mode-Level vs. Implementation-Level Physical Security in Symmetric Cryptography: A Practical Guide Through the Leakage-Resistance Jungle", Advances in Cryptology - CRYPTO 2020, Lecture Notes in Computer Science, vol. 12170, pp. 369-400, DOI 10.1007/978-3-030-56784-2_13, 2020, <https://doi.org/10.1007/978-3-030-56784-2_13>.

[BBCLNSS21] Bhaumik, R., Bonnetain, X., Chailloux, A., Leurent, G., Naya-Plasencia, M., Schrottenloher, A., and Y. Seurin, "QCB: Efficient Quantum-Secure Authenticated Encryption", Advances in Cryptology - ASIACRYPT 2021, Lecture Notes in Computer Science, vol. 13090, pp. 668-698, DOI 10.1007/978-3-030-92062-3_23, 2021, <https://doi.org/10.1007/978-3-030-92062-3_23>.

[BC09] Black, J. and M. Cochran, "MAC Reforgeability", Fast Software Encryption (FSE 2009), Lecture Notes in Computer Science, vol. 5665, pp. 345-362, DOI 10.1007/978-3-642-03317-9_21, 2009, <https://doi.org/10.1007/978-3-642-03317-9_21>.

[BH22] Bellare, M. and V.T. Hoang, "Efficient Schemes for Committing Authenticated Encryption", Advances in Cryptology - EUROCRYPT 2022, Lecture Notes in Computer Science, vol. 13276, pp. 845-875, DOI 10.1007/978-3-031-07085-3_29, 2022, <https://doi.org/10.1007/978-3-031-07085-3_29>.

[BHT18] Bose, P., Hoang, V.T., and S. Tessaro, "Revisiting AES-GCM-SIV: Multi-user Security, Faster Key Derivation, and Better Bounds", Advances in Cryptology - EUROCRYPT 2018, Lecture Notes in Computer Science, vol. 10820, pp. 468-499, DOI 10.1007/978-3-319-78381-9_18, 2018, <https://doi.org/10.1007/978-3-319-78381-9_18>.

[BKY02] Buonanno, E., Katz, J., and M. Yung, "Incremental Unforgeable Encryption", Fast Software Encryption (FSE 2001), Lecture Notes in Computer Science, vol. 2355, pp. 109-124, DOI 10.1007/3-540-45473-X_9, 2002, <https://doi.org/10.1007/3-540-45473-X_9>.

[BM18] Barbosa, M. and P. Farshim, "Indifferentiable Authenticated Encryption", Advances in Cryptology - CRYPTO 2018, Lecture Notes in Computer Science, vol. 10991, pp. 187-220, DOI 10.1007/978-3-319-96884-1_7, 2018, <https://doi.org/10.1007/978-3-319-96884-1_7>.

[BMOS17] Barwell, G., Martin, D.P., Oswald, E., and M. Stam, "Authenticated Encryption in the Face of Protocol and Side Channel Leakage", Advances in Cryptology - ASIACRYPT 2017, Lecture Notes in Computer Science, vol. 10624, pp. 693-723, DOI 10.1007/978-3-319-70694-8_24, 2017, <https://doi.org/10.1007/978-3-319-70694-8_24>.

[BN08] Bellare, M. and C. Namprempre, "Authenticated Encryption: Relations among Notions and Analysis of the Generic Composition Paradigm", Journal of Cryptology, vol. 21, pp. 469-491, DOI 10.1007/s00145-008-9026-x, 2008, <https://doi.org/10.1007/s00145-008-9026-x>.

[BNT19] Bellare, M., Ng, R., and B. Tackmann, "Nonces Are Noticed: AEAD Revisited", Advances in Cryptology - CRYPTO 2019, Lecture Notes in Computer Science, vol. 11692, pp. 235-265, DOI 10.1007/978-3-030-26948-7_9, 2019, <https://doi.org/10.1007/978-3-030-26948-7_9>.

[BPPS17] Berti, F., Pereira, O., Peters, T., and F.-X. Standaert, "On Leakage-Resilient Authenticated Encryption with Decryption Leakages", IACR Transactions on Symmetric Cryptology, vol. 2017, no. 3, pp. 271-293, DOI 10.13154/tosc.v2017.i3.271-293, 2017, <https://doi.org/10.13154/tosc.v2017.i3.271-293>.

[BT16] Bellare, M. and B. Tackmann, "The Multi-user Security of Authenticated Encryption: AES-GCM in TLS 1.3", Advances in Cryptology - CRYPTO 2016, Lecture Notes in Computer Science, vol. 9814, pp. 247-276, DOI 10.1007/978-3-662-53018-4_10, 2016, <https://doi.org/10.1007/978-3-662-53018-4_10>.

[C20] Chakraborti, A., Datta, N., Jha, A., Mancillas-López, C., Nandi, M., and Y. Sasaki, "INT-RUP Secure Lightweight Parallel AE Modes", IACR Transactions on Symmetric Cryptology, vol. 2019, no.4, pp. 81-118, DOI 10.13154/tosc.v2019.i4.81-118, 2020, <https://doi.org/10.13154/tosc.v2019.i4.81-118>.

[CJRR99] Chari, S., Jutla, C.S., Rao, J.R., and P. Rohatgi, "Towards Sound Approaches to Counteract Power-Analysis Attacks", Advances in Cryptology - CRYPTO'99, Lecture Notes in Computer Science, vol. 1666, pp. 398-412, DOI 10.1007/3-540-48405-1_26, 1999, <https://doi.org/10.1007/3-540-48405-1_26>.

[CR22] Chan, J. and P. Rogaway, "On Committing Authenticated-Encryption", Computer Security - ESORICS 2022, Lecture Notes in Computer Science, vol. 13555, pp. 275-294, DOI 10.1007/978-3-031-17146-8_14, 2022, <https://doi.org/10.1007/978-3-031-17146-8_14>.

[DEMS21a] Dobraunig, C., Eichlseder, M., Mendel, F., and M. Schläffer, "Ascon v1.2: Lightweight Authenticated Encryption and Hashing", Journal of Cryptology, vol. 34, no. 33, DOI 10.1007/s00145-021-09398-9, 2021, <https://doi.org/10.1007/s00145-021-09398-9>.

[DEMS21b] Dobraunig, C., Eichlseder, M., Mendel, F., and M. Schläffer, "Ascon v1.2", Submission to the NIST LWC Competition, 2021.

[DGGP21] Degabriele, J.P., Govinden, J., Günther, F., and K. Paterson, "The Security of ChaCha20-Poly1305 in the Multi-User Setting", Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security (CCS '21), pp. 1981-2003, DOI 10.1145/3460120.3484814, 2021, <https://doi.org/10.1145/3460120.3484814>.

[EV17] Endignoux, G. and D. Vizár, "Linking Online Misuse-Resistant Authenticated Encryption and Blockwise Attack Models", IACR Transactions on Symmetric Cryptology, vol. 2016, no. 2, pp. 125-144, DOI 10.13154/TOSC.V2016.I2.125-144, 2017, <https://doi.org/10.13154/TOSC.V2016.I2.125-144>.

[FFL12] Fleischmann, E., Forler, C., and S. Lucks, "McOE: A Family of Almost Foolproof On-Line Authenticated Encryption Schemes", Fast Software Encryption (FSE 2012), Lecture Notes in Computer Science, vol. 7549, pp. 196-215, DOI 10.1007/978-3-642-34047-5_12, 2012, <https://doi.org/10.1007/978-3-642-34047-5_12>.

[FJMV2004] Fouque, P.-A., Joux, A., Martinet, G., and F. Valette, "Authenticated On-Line Encryption", Selected Areas in Cryptography (SAC 2003), Lecture Notes in Computer Science, vol. 3006, DOI 10.1007/978-3-540-24654-1_11, 2004, <https://doi.org/10.1007/978-3-540-24654-1_11>.

[FLLW17] Forler, C., List, E., Lucks, S., and J. Wenzel, "Reforgeability of Authenticated Encryption Schemes", Information Security and Privacy (ACISP 2017), Lecture Notes in Computer Science, vol. 10343, pp. 19-37, DOI 10.1007/978-3-319-59870-3_2, 2017, <https://doi.org/10.1007/978-3-319-59870-3_2>.

[FOR17] Farshim, P., Orlandi, C., and R. Rosie, "Security of Symmetric Primitives under Incorrect Usage of Keys", IACR Transactions on Symmetric Cryptology, vol. 2017, no. 1, pp. 449-473, DOI 10.13154/tosc.v2017.i1.449-473, 2017, <https://doi.org/10.13154/tosc.v2017.i1.449-473>.

[G17] Gagliardoni, T., "Quantum Security of Cryptographic Primitives", Ph.D. Thesis, Technische Universität Darmstadt, 2017, <https://tuprints.ulb.tu-darmstadt.de/6019/>.

[GLR17] Grubbs, P., Lu, J., and T. Ristenpart, "Message Franking via Committing Authenticated Encryption", Advances in Cryptology - CRYPTO 2017, Lecture Notes in Computer Science, vol. 10403, pp. 66-97, DOI 10.1007/978-3-319-63697-9_3, 2017, <https://doi.org/10.1007/978-3-319-63697-9_3>.

[GPPS19] Guo, C., Pereira, O., Peters, T., and F.-X. Standaert, "Authenticated Encryption with Nonce Misuse and Physical Leakages: Definitions, Separation Results and First Construction", Progress in Cryptology - LATINCRYPT 2019, Lecture Notes in Computer Science, vol. 11774, pp. 150-172, DOI 10.1007/978-3-030-30530-7_8, 2019, <https://doi.org/10.1007/978-3-030-30530-7_8>.

[HKR2015] Hoang, V.T., Krovetz, T., and P. Rogaway, "Robust Authenticated-Encryption AEZ and the Problem That It Solves", Advances in Cryptology -- EUROCRYPT 2015, Lecture Notes in Computer Science, vol. 9056, pp. 15-44, DOI 10.1007/978-3-662-46800-5_2, 2015, <https://doi.org/10.1007/978-3-662-46800-5_2>.

[HRRV15] Hoang, VT., Reyhanitabar, R., Rogaway, P., and D. Vizár, "Online Authenticated-Encryption and its Nonce-Reuse Misuse-Resistance", Advances in Cryptology -- CRYPTO 2015, Lecture Notes in Computer Science, vol. 9215, pp. 493-517, DOI 10.1007/978-3-662-47989-6_24, 2015, <https://doi.org/10.1007/978-3-662-47989-6_24>.

[HTT18] Hoang, V.T., Tessaro, S., and A. Thiruvengadam, "The Multi-user Security of GCM, Revisited: Tight Bounds for Nonce Randomization", Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security (CCS '18), pp. 1429-1440, DOI 10.1145/3243734.3243816, 2018, <https://doi.org/10.1145/3243734.3243816>.

[IIM25] Inoue, A., Iwata, T., and K. Minematsu, "Comprehensive Robustness Analysis of GCM, CCM, and OCB3", Topics in Cryptology - CT-RSA 2025, Lecture Notes in Computer Science, vol. 15598, DOI 10.1007/978-3-031-88661-4_4, 2025, <https://doi.org/10.1007/978-3-031-88661-4_4>.

[JMV2002] Joux, A., Martinet, G., and F. Valette, "Blockwise-Adaptive Attackers Revisiting the (In)Security of Some Provably Secure Encryption Modes: CBC, GEM, IACBC", Advances in Cryptology - CRYPTO 2002, Lecture Notes in Computer Science, vol. 2442, DOI 10.1007/3-540-45708-9_2, 2002, <https://doi.org/10.1007/3-540-45708-9_2>.

[JNPS21] Jean, M., Nikolić, I., Peyrin, T., and Y. Seurin, "The Deoxys AEAD family", Journal of Cryptology, vol. 34, no. 31, DOI 10.1007/s00145-021-09397-w, 2021, <https://doi.org/10.1007/s00145-021-09397-w>.

[KLLNP16] Menda, M., Leurent, G., Leverrier, A., and M. Naya-Plasencia, "Quantum Differential and Linear Cryptanalysis", IACR Transactions on Symmetric Cryptology, vol. 2016, no.1, pp. 71-94, DOI 10.13154/tosc.v2016.i1.71-94, 2016, <https://doi.org/10.13154/tosc.v2016.i1.71-94>.

[LGR21] Len, J., Grubbs, P., and T. Ristenpart, "Partitioning Oracle Attacks", 30th USENIX Security Symposium (USENIX Security 21), pp. 195-212, 2021.

[LMP17] Luykx, A., Mennink, B., and K. Paterson, "Analyzing Multi-key Security Degradation", Advances in Cryptology - ASIACRYPT 2017, Lecture Notes in Computer Science, vol. 10625, pp. 575-605, DOI 10.1007/978-3-319-70697-9_20, 2017, <https://doi.org/10.1007/978-3-319-70697-9_20>.

[M05] McGrew, D., "Efficient authentication of large, dynamic data sets using Galois/Counter Mode (GCM)", Third IEEE International Security in Storage Workshop (SISW'05), pp. 6-94, DOI 10.1109/SISW.2005.3, 2005, <https://doi.org/10.1109/SISW.2005.3>.

[MBTM17] McKay, K., Bassham, L., Turan, MS., and N. Mouha, "Report on Lightweight Cryptography", NISTIR 8114, DOI 10.6028/NIST.IR.8114, 2017, <https://doi.org/10.6028/NIST.IR.8114>.

[MLGR23] Menda, S., Julia, J., Grubbs, P., and T. Ristenpart, "Context Discovery and Commitment Attacks: How to Break CCM, EAX, SIV, and More", Advances in Cryptology - EUROCRYPT 2023, Lecture Notes in Computer Science, vol. 14007, pp. 379-407, DOI 10.1007/978-3-031-30634-1_13, 2023, <https://doi.org/10.1007/978-3-031-30634-1_13>.

[R02] Rogaway, P., "Authenticated-encryption with associated-data", Proceedings of the 9th ACM Conference on Computer and Communications Security (CCS '02), pp. 98-107, DOI 10.1145/586110.586125, 2002, <https://doi.org/10.1145/586110.586125>.

[RFC4303] Kent, S., "IP Encapsulating Security Payload (ESP)", RFC 4303, DOI 10.17487/RFC4303, December 2005, <https://www.rfc-editor.org/info/rfc4303>.

[RFC7253] Krovetz, T. and P. Rogaway, "The OCB Authenticated-Encryption Algorithm", RFC 7253, DOI 10.17487/RFC7253, May 2014, <https://www.rfc-editor.org/info/rfc7253>.

[RFC8221] Wouters, P., Migault, D., Mattsson, J., Nir, Y., and T. Kivinen, "Cryptographic Algorithm Implementation Requirements and Usage Guidance for Encapsulating Security Payload (ESP) and Authentication Header (AH)", RFC 8221, DOI 10.17487/RFC8221, October 2017, <https://www.rfc-editor.org/info/rfc8221>.

[RFC8439] Nir, Y. and A. Langley, "ChaCha20 and Poly1305 for IETF Protocols", RFC 8439, DOI 10.17487/RFC8439, June 2018, <https://www.rfc-editor.org/info/rfc8439>.

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/info/rfc8446>.

[RFC8452] Gueron, S., Langley, A., and Y. Lindell, "AES-GCM-SIV: Nonce Misuse-Resistant Authenticated Encryption", RFC 8452, DOI 10.17487/RFC8452, April 2019, <https://www.rfc-editor.org/info/rfc8452>.

[RFC8645] Smyshlyaev, S., Ed., "Re-keying Mechanisms for Symmetric Keys", RFC 8645, DOI 10.17487/RFC8645, August 2019, <https://www.rfc-editor.org/info/rfc8645>.

[RFC9000] Iyengar, J., Ed. and M. Thomson, Ed., "QUIC: A UDP-Based Multiplexed and Secure Transport", RFC 9000, DOI 10.17487/RFC9000, May 2021, <https://www.rfc-editor.org/info/rfc9000>.

[RFC9058] Smyshlyaev, S., Ed., Nozdrunov, V., Shishkin, V., and E. Griboedova, "Multilinear Galois Mode (MGM)", RFC 9058, DOI 10.17487/RFC9058, June 2021, <https://www.rfc-editor.org/info/rfc9058>.

[RS06] Rogaway, R. and T. Shrimpton, "A Provable-Security Treatment of the Key-Wrap Problem", Advances in Cryptology - EUROCRYPT 2006, Lecture Notes in Computer Science, vol. 4004, pp. 373-390, DOI 10.1007/11761679_23, 2016, <https://doi.org/10.1007/11761679_23>.

[S04] Shrimpton, T., "A Characterization of Authenticated-Encryption as a Form of Chosen-Ciphertext Security", Cryptology ePrint Archive, Paper 2004/272, 2004, <https://eprint.iacr.org/2004/272>.

[SY16] Sasaki, Y. and K. Yasuda, "A New Mode of Operation for Incremental Authenticated Encryption with Associated Data", Selected Areas in Cryptography - SAC 2015, Lecture Notes in Computer Science, vol. 9566, pp. 397-416, DOI 10.1007/978-3-319-31301-6_23, 2016, <https://doi.org/10.1007/978-3-319-31301-6_23>.

[YSS23] Naito, Y., Sasaki, Y., and T. Sugawara, "Committing Security of Ascon: Cryptanalysis on Primitive and Proof on Mode", IACR Transactions on Symmetric Cryptology, vol. 2023, no. 4, pp. 420-451, DOI 10.46586/tosc.v2023.i4.420-451, 2023, <https://doi.org/10.46586/tosc.v2023.i4.420-451>.

Appendix A. AEAD Algorithms with Additional Functionality

In this section, we briefly discuss AEAD algorithms that provide additional functionality. As noted in Section 4.1, each additional functionality requires a redefinition of the conventional AEAD interface; thus, each additional functionality property defines a new class of cryptographic algorithms.

Most importantly, for every AEAD class with additional functionality, conventional security properties must be redefined concerning the targeted additional functionality and the new interface. Although it might be possible to consider a particular AEAD algorithm with additional functionality as a conventional AEAD algorithm and study it for the conventional confidentiality and integrity, security (or insecurity) in that sense won't be sufficient to label that algorithm as a secure (or insecure) additional functionality AEAD. Only security in the sense of the redefined conventional properties would suffice.

For the examples given in this section, we leave it out of scope how to concretely redefine conventional security for these classes; we only briefly describe the additional functionality they offer and provide further references.

A.1. Incremental Authenticated Encryption

Definition:
: For a message that only partly differs from some previous message, an AEAD algorithm allows re-encrypting and authenticating that message (associated data and a plaintext pair) faster than processing it from scratch.

Examples:
: Incremental AEAD algorithm of [SY16]

Security notions:
: Privacy, authenticity [SY16]

Notes:
: When compared with conventional AEAD, the interface of an incremental AEAD algorithm is usually expanded with several operations, which perform different types of updates. For example, one can consider operations such as "Append" or "Chop", which provide a straightforward additional functionality. A comprehensive definition of an incremental AEAD interface is provided in [SY16].

Further reading:
: [SY16], [M05], [BKY02]

A.2. Robust Authenticated Encryption

Definition:
: An AEAD algorithm allows users to choose a desired ciphertext expansion (the difference between the length of plaintext and corresponding ciphertext) along with an input to the encryption operation. This feature enables the regulation of desired data integrity guarantees, which depend on ciphertext expansion, for each particular application while using the same algorithm implementation.

Examples:
: AEZ [HKR2015]

Security notions:
: Robust Authenticated Encryption (RAE) [HKR2015]

Notes:
: The security goal of robust AEAD algorithms is to ensure the best possible security, even with small ciphertext expansion (referred to as stretch). For instance, analyzing any AEAD algorithm with a one-byte stretch for conventional integrity reveals insecurity, as the

probability of forging a ciphertext is no less than 1/256. Nonetheless, from the robust AEAD perspective, an algorithm with such forgery probability for a one-byte ciphertext expansion is secure, representing the best achievable security in that scenario.

Further reading:
[HKR2015]

Acknowledgments

This document benefited greatly from the comments received from the CFRG community, for which we are very grateful. We would also like to extend special appreciation to Liliya Akhmetzyanova, Evgeny Alekseev, Alexandra Babueva, Frank Denis, Kirill Kutsenok, Sergey Kyazhin, Samuel Lucas, Grigory Marshalko, Christopher Patton, and Christopher Wood for their thoughtful comments, proposals, and discussions.

Author's Address

Andrey Bozhko (EDITOR)
CryptoPro
Email: andbogc@gmail.com