Authors: F. Günther   M. Thomson   C.A. Wood
         ETH Zurich   Mozilla   Cloudflare

Usage Limits on AEAD Algorithms

Abstract

An Authenticated Encryption with Associated Data (AEAD) algorithm
provides confidentiality and integrity. Excessive use of the same
key can give an attacker advantages in breaking these properties.
This document provides simple guidance for users of common AEAD
functions about how to limit the use of keys in order to bound the
advantage given to an attacker.

Discussion Venues

This note is to be removed before publishing as an RFC.

Source for this draft and an issue tracker can be found at https://
github.com/chris-wood/draft-wood-cfrg-aead-limits.

Status of This Memo

This Internet-Draft is submitted in full conformance with the
provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering
Task Force (IETF). Note that other groups may also distribute
working documents as Internet-Drafts. The list of current Internet-
Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six
months and may be updated, replaced, or obsoleted by other documents
at any time. It is inappropriate to use Internet-Drafts as reference
material or to cite them other than as "work in progress."

This Internet-Draft will expire on 21 November 2020.

Copyright Notice

Table of Contents

1. Introduction

An Authenticated Encryption with Associated Data (AEAD) algorithm
provides confidentiality and integrity. [RFC5116] specifies an AEAD
as a function with four inputs - secret key, nonce, plaintext, and
optional associated data - that produces ciphertext output and error
code indicating success or failure. The ciphertext is typically
composed of the encrypted plaintext bytes and an authentication tag.

The generic AEAD interface does not describe usage limits. Each AEAD
algorithm does describe limits on its inputs, but these are
formulated as strict functional limits, such as the maximum length
of inputs, which are determined by the properties of the underlying
AEAD composition. Degradation of the security of the AEAD as a
single key is used multiple times is not given a thorough treatment.

The number of times a single pair of key and nonce can be used might
also be relevant to security. For some algorithms, such as
AEAD_AES_128_GCM or AEAD_AES_256_GCM, this limit is 1 and using the
same pair of key and nonce has serious consequences for both
confidentiality and integrity; see [NonceDisrespecting]. Nonce-reuse
resistant algorithms like AEAD_AES_128_GCM_SIV can tolerate a
limited amount of nonce reuse.

It is good practice to have limits on how many times the same key
(or pair of key and nonce) are used. Setting a limit based on some
measurable property of the usage, such as number of protected
messages or amount of data transferred, ensures that it is easy to
apply limits. This might require the application of simplifying
assumptions. For example, TLS 1.3 specifies limits on the number of
records that can be protected, using the simplifying assumption that
records are the same size; see Section 5.5 of [TLS].

Currently, AEAD limits and usage requirements are scattered among
peer-reviewed papers, standards documents, and other RFCs.
Determining the correct limits for a given setting is challenging as
papers do not use consistent labels or conventions, and rarely apply
any simplifications that might aid in reaching a simple limit.

The intent of this document is to collate all relevant information
about the proper usage and limits of AEAD algorithms in one place.
This may serve as a standard reference when considering which AEAD
algorithm to use, and how to use it.

2. Requirements Notation

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. Notation

This document defines limitations in part using the quantities below.

Symbol	Description
n	Size of the AEAD block cipher (in bits)
t	Size of the authentication tag (in bits)
l	Length of each message (in blocks)
s	Total plaintext length in all messages (in blocks)
q	Number of encryption attempts
v	Number of forgery attempts
p	Adversary attack probability

Table 1

For each AEAD algorithm, we define the confidentiality and integrity advantage roughly as the advantage an attacker has in breaking the corresponding security property for the algorithm. Specifically:

  *Confidentiality advantage (CA): The advantage of an attacker
   succeeding in breaking the confidentiality properties of the
   AEAD. In this document, the definition of confidentiality
   advantage is the increase in the probability that an attacker is
   able to successfully distinguish an AEAD ciphertext from the
   output of an ideal pseudorandom permutation (PRP).

  *Integrity advantage (IA): The probability of an attacker
   succeeding in breaking the integrity properties of the AEAD. In
   this document, the definition of integrity advantage is the
   probability that an attacker is able to forge a ciphertext that
   will be accepted as valid.

Each application requires a different application of limits in order to keep CA and IA sufficiently small. For instance, TLS aims to keep CA below 2^{-60} and IA below 2^{-57}. See [TLS], Section 5.5.

4. Calculating Limits

Once an upper bound on CA and IA are determined, this document defines a process for determining two overall limits:

  *Confidentiality limit (CL): The number of bytes of plaintext and
   maybe authenticated additional data (AAD) an application can
   encrypt before giving the adversary a non-negligible CA.

*Integrity limit (IL): The number of bytes of ciphertext and maybe
 authenticated additional data (AAD) an application can process,
 either successfully or not, before giving the adversary a non-
 negligible IA.

For an AEAD based on a block function, it is common for these limits
to be expressed instead in terms of the number of blocks rather than
bytes. Furthermore, it might be more appropriate to track the number
of messages rather than track bytes. Therefore, the guidance is
usually based on the total number of blocks processed (s). To aid in
calculating limits for message-based protocols, a formulation of
limits that includes a maximum message size (l) is included.

All limits are based on the total number of messages, either the
number of protected messages (q) or the number of forgery attempts
(v); which correspond to CL and IL respectively.

Limits are then derived from those bounds using a target attacker
probability. For example, given a confidentiality advantage of v *
(8l / 2^106) and attacker success probability of p, the algorithm
remains secure, i.e., the adversary's advantage does not exceed the
probability of success, provided that v <= (p * 2^106) / 8l. In
turn, this implies that v <= (p * 2^106) / 8l is the corresponding
limit.

5. AEAD Limits and Requirements

This section summarizes the confidentiality and integrity bounds and
limits for modern AEAD algorithms used in IETF protocols, including:
AEAD_AES_128_GCM [RFC5116], AEAD_AES_256_GCM [RFC5116],
AEAD_AES_128_CCM [RFC5116], AEAD_CHACHA20_POLY1305 [RFC8439],
AEAD_AES_128_CCM_8 [RFC6655].

The CL and IL values bound the total number of encryption and
forgery queries (q and v). Alongside each value, we also specify
these bounds.

5.1. AEAD_AES_128_GCM and AEAD_AES_256_GCM

The CL and IL values for AES-GCM are derived in [AEBounds] and
summarized below. For this AEAD, n = 128 and t = 128 [GCM]. In this
example, the length s is the sum of AAD and plaintext, as described
in [GCMProofs].

5.1.1. Confidentiality Limit

CA = ((s + q + 1)^2) / 2^129

This implies the following usage limit:

```
q + s <= p^(1/2) * 2^(129/2) - 1
```

   Which, for a message-based protocol with s <= q * l, if we assume
   that every packet is size l, produces the limit:

```
q <= (p^(1/2) * 2^(129/2) - 1) / (l + 1)
```

5.1.2. Integrity Limit

```
IA = 2 * (v * (l + 1)) / 2^128
```

   This implies the following limit:

```
v <= (p * 2^127) / (l + 1)
```

5.2. AEAD_CHACHA20_POLY1305

   The only known analysis for AEAD_CHACHA20_POLY1305
   [ChaCha20Poly1305Bounds] combines the confidentiality and integrity
   limits into a single expression, covered below:

```
CA = IA = v * (8l / 2^106)
```

   This advantage is a tight reduction based on the underlying Poly1305
   PRF [Poly1305]. It implies the following limit:

```
v <= (p * 2^106) / 8l
```

5.3. AEAD_AES_128_CCM

   The CL and IL values for AEAD_AES_128_CCM are derived from [CCM-
   ANALYSIS] and specified in the QUIC-TLS mapping specification [I-
   D.ietf-quic-tls]. This analysis uses the total number of underlying
   block cipher operations to derive its bound. For CCM, this number is
   the sum of: the length of the associated data in blocks, the length
   of the ciphertext in blocks, the length of the plaintext in blocks,
   plus 1.

   In the following limits, this is simplified to a value of twice the
   length of the packet in blocks, i.e., 2l represents the effective
   length, in number of block cipher operations, of a message with l
   blocks. This simplification is based on the observation that common
   applications of this AEAD carry only a small amount of associated
   data compared to ciphertext. For example, QUIC has 1 to 3 blocks of
   AAD.

   For this AEAD, n = 128 and t = 128.

5.3.1. Confidentiality Limit

```
CA = (2l * q)^2 / 2^n
   = (2l * q)^2 / 2^128
```

This implies the following limit:

```
q <= sqrt((p * (2^127)) / l^2)
```

5.3.2. Integrity Limit

```
IA = v / 2^t + (2l * (v + q))^2 / 2^n
   = v / 2^128 + (2l * (v + q))^2 / 2^128
```

This implies the following limit:

```
v + (2l * (v + q))^2 <= p * 2^128
```

In a setting where v or q is sufficiently large, v is negligible
compared to (2l * (v + q))^2, so this this can be simplified to:

```
v + q <= p^(1/2) * 2^63 / l
```

5.4. AEAD_AES_128_CCM_8

The analysis in [CCM-ANALYSIS] also applies to this AEAD, but the
reduced tag length of 64 bits changes the integrity limit
calculation considerably.

```
IA = v / 2^t + (2l * (v + q))^2 / 2^n
   = v / 2^64 + (2l * (v + q))^2 / 2^128
```

This results in reducing the limit on v by a factor of 2^64.

```
v * 2^64 + (2l * (v + q))^2 <= p * 2^128
```

6. Security Considerations

Many of the formulae in this document depend on simplifying
assumptions that are not universally applicable. When using this
document to set limits, it is necessary to validate all these
assumptions for the setting in which the limits might apply. In most
cases, the goal is to use assumptions that result in setting a more
conservative limit, but this is not always the case.

7. IANA Considerations

This document does not make any request of IANA.

8. References

8.1. Normative References

[AEBounds] Luykx, A. and K. Paterson, "Limits on Authenticated
            Encryption Use in TLS", 8 March 2016, <http://
            www.isg.rhul.ac.uk/~kp/TLS-AEbounds.pdf>.

[CCM-ANALYSIS] Jonsson, J., "On the Security of CTR + CBC-MAC", DOI
            10.1007/3-540-36492-7_7, Selected Areas in Cryptography
            pp. 76-93, 2003, <https://doi.org/
            10.1007/3-540-36492-7_7>.

[ChaCha20Poly1305Bounds] Procter, G., "A Security Analysis of the
            Composition of ChaCha20 and Poly1305", 2014, <https://
            eprint.iacr.org/2014/613.pdf>.

[GCM]   Dworkin, M., "Recommendation for Block Cipher Modes of
            Operation: Galois/Counter Mode (GCM) and GMAC", NIST
            Special Publication 800-38D, November 2007.

[GCMProofs] Iwata, T., Ohashi, K., and K. Minematsu, "Breaking and
            Repairing GCM Security Proofs", n.d., <https://
            eprint.iacr.org/2012/438.pdf>.

[Poly1305] Bernstein, D.J., "The Poly1305-AES message-authentication
            code", International Workshop on Fast Software
            Encryption, 2005 , n.d., <https://link.springer.com/
            content/pdf/10.1007/11502760_3.pdf>.

[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>.

[RFC6655]   McGrew, D. and D. Bailey, "AES-CCM Cipher Suites for
            Transport Layer Security (TLS)", RFC 6655, DOI 10.17487/

            RFC6655, July 2012, <https://www.rfc-editor.org/info/
            rfc6655>.

   [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>.

   [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>.

8.2. Informative References

   [I-D.ietf-quic-tls] Thomson, M. and S. Turner, "Using TLS to Secure
            QUIC", Work in Progress, Internet-Draft, draft-ietf-quic-
            tls-28, 19 May 2020, <http://www.ietf.org/internet-
            drafts/draft-ietf-quic-tls-28.txt>.

   [NonceDisrespecting] Bock, H., Zauner, A., Devlin, S., Somorovsky,
            J., and P. Jovanovic, "Nonce-Disrespecting Adversaries --
            Practical Forgery Attacks on GCM in TLS", 17 May 2016,
            <https://eprint.iacr.org/2016/475.pdf>.

   [TLS]   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>.

Authors' Addresses

   Felix Günther
   ETH Zurich

   Email: mail@felixguenther.info

   Martin Thomson
   Mozilla

   Email: mt@lowentropy.net

   Christopher A. Wood
   Cloudflare

   Email: caw@heapingbits.net