Network Working Group
Request for Comments: 4309
Category: Standards Track
December 2005

R. Housley
Vigil Security

Using Advanced Encryption Standard (AES) CCM Mode with IPsec Encapsulating Security Payload (ESP)

Status of This Memo

Copyright Notice

Abstract

This document describes the use of Advanced Encryption Standard (AES) in Counter with CBC-MAC (CCM) Mode, with an explicit initialization vector (IV), as an IPsec Encapsulating Security Payload (ESP) mechanism to provide confidentiality, data origin authentication, and connectionless integrity.

Table of Contents

1. Introduction

The Advanced Encryption Standard (AES) [AES] is a block cipher, and it can be used in many different modes. This document describes the use of AES in CCM (Counter with CBC-MAC) mode (AES CCM), with an explicit initialization vector (IV), as an IPsec Encapsulating Security Payload (ESP) [ESP] mechanism to provide confidentiality, data origin authentication, and connectionless integrity.

This document does not provide an overview of IPsec. However, information about how the various components of IPsec and the way in which they collectively provide security services is available in [ARCH] and [ROAD].

1.1. Conventions Used in This Document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [STDWORDS].

2. AES CCM Mode

CCM is a generic authenticate-and-encrypt block cipher mode [CCM]. In this specification, CCM is used with the AES [AES] block cipher.

AES CCM has two parameters:

M M indicates the size of the integrity check value (ICV). CCM defines values of 4, 6, 8, 10, 12, 14, and 16 octets; However, to maintain alignment and provide adequate security, only the values that are a multiple of four and are at least eight are permitted. Implementations MUST support M values of 8 octets and 16 octets, and implementations MAY support an M value of 12 octets.

L L indicates the size of the length field in octets. CCM defines values of L between 2 octets and 8 octets. This specification only supports L = 4. Implementations MUST support an L value of 4 octets, which accommodates a full Jumbogram [JUMBO]; however, the length includes all of the encrypted data, which also includes the ESP Padding, Pad Length, and Next Header fields.

There are four inputs to CCM originator processing:

key
A single key is used to calculate the ICV using CBC-MAC and to perform payload encryption using counter mode. AES supports key sizes of 128 bits, 192 bits, and 256 bits. The default key

size is 128 bits, and implementations MUST support this key size. Implementations MAY also support key sizes of 192 bits and 256 bits.

nonce
The size of the nonce depends on the value selected for the parameter L. It is 15-L octets. Implementations MUST support a nonce of 11 octets. The construction of the nonce is described in Section 4.

payload
The payload of the ESP packet. The payload MUST NOT be longer than 4,294,967,295 octets, which is the maximum size of a Jumbogram [JUMBO]; however, the ESP Padding, Pad Length, and Next Header fields are also part of the payload.

AAD
CCM provides data integrity and data origin authentication for some data outside the payload. CCM does not allow additional authenticated data (AAD) to be longer than 18,446,744,073,709,551,615 octets. The ICV is computed from the ESP header, Payload, and ESP trailer fields, which is significantly smaller than the CCM-imposed limit. The construction of the AAD described in Section 5.

AES CCM requires the encryptor to generate a unique per-packet value and to communicate this value to the decryptor. This per-packet value is one of the component parts of the nonce, and it is referred to as the initialization vector (IV). The same IV and key combination MUST NOT be used more than once. The encryptor can generate the IV in any manner that ensures uniqueness. Common approaches to IV generation include incrementing a counter for each packet and linear feedback shift registers (LFSRs).

AES CCM employs counter mode for encryption. As with any stream cipher, reuse of the same IV value with the same key is catastrophic. An IV collision immediately leaks information about the plaintext in both packets. For this reason, it is inappropriate to use this CCM with statically configured keys. Extraordinary measures would be needed to prevent reuse of an IV value with the static key across power cycles. To be safe, implementations MUST use fresh keys with AES CCM. The Internet Key Exchange (IKE) [IKE] protocol or IKEv2 [IKEv2] can be used to establish fresh keys.

3. ESP Payload

The ESP payload is composed of the IV followed by the ciphertext. The payload field, as defined in [ESP], is structured as shown in Figure 1.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                     Initialization Vector                     |
   |                            (8 octets)                         |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                                                               |
   ~                  Encrypted Payload (variable)                 ~
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                                                               |
   ~                 Authentication Data (variable)                ~
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 1. ESP Payload Encrypted with AES CCM

3.1. Initialization Vector (IV)

The AES CCM IV field MUST be eight octets. The IV MUST be chosen by the encryptor in a manner that ensures that the same IV value is used only once for a given key. The encryptor can generate the IV in any manner that ensures uniqueness. Common approaches to IV generation include incrementing a counter for each packet and linear feedback shift registers (LFSRs).

Including the IV in each packet ensures that the decryptor can generate the key stream needed for decryption, even when some datagrams are lost or reordered.

3.2. Encrypted Payload

The encrypted payload contains the ciphertext.

AES CCM mode does not require plaintext padding. However, ESP does require padding to 32-bit word-align the authentication data. The Padding, Pad Length, and Next Header fields MUST be concatenated with the plaintext before performing encryption, as described in [ESP]. When padding is required, it MUST be generated and checked in accordance with the conventions specified in [ESP].

3.3. Authentication Data

AES CCM provides an encrypted ICV. The ICV provided by CCM is carried in the Authentication Data fields without further encryption. Implementations MUST support ICV sizes of 8 octets and 16 octets. Implementations MAY also support ICV 12 octets.

4. Nonce Format

Each packet conveys the IV that is necessary to construct the sequence of counter blocks used by counter mode to generate the key stream. The AES counter block is 16 octets. One octet is used for the CCM Flags, and 4 octets are used for the block counter, as specified by the CCM L parameter. The remaining octets are the nonce. These octets occupy the second through the twelfth octets in the counter block. Figure 2 shows the format of the nonce.

```
       0                   1                   2                   3
       0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
                      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
                      |                  Salt                         |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |                     Initialization Vector                     |
      |                                                               |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2. Nonce Format

The components of the nonce are as follows:

Salt The salt field is 24 bits. As the name implies, it contains an unpredictable value. It MUST be assigned at the beginning of the security association. The salt value need not be secret, but it MUST NOT be predictable prior to the beginning of the security association.

Initialization Vector The IV field is 64 bits. As described in Section 3.1, the IV MUST be chosen by the encryptor in a manner that ensures that the same IV value is used only once for a given key.

This construction permits each packet to consist of up to:

```
      2^32 blocks  =  4,294,967,296 blocks
                   = 68,719,476,736 octets
```

This construction provides more key stream for each packet than is needed to handle any IPv6 Jumbogram [JUMBO].

5. AAD Construction

The data integrity and data origin authentication for the Security Parameters Index (SPI) and (Extended) Sequence Number fields is provided without encrypting them. Two formats are defined: one for 32-bit sequence numbers and one for 64-bit extended sequence numbers. The format with 32-bit sequence numbers is shown in Figure 3, and the format with 64-bit extended sequence numbers is shown in Figure 4.

Sequence Numbers are conveyed canonical network byte order. Extended Sequence Numbers are conveyed canonical network byte order, placing the high-order 32 bits first and the low-order 32 bits second. Canonical network byte order is fully described in RFC 791, Appendix B.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                               SPI                             |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                     32-bit Sequence Number                    |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 3. AAD Format with 32-bit Sequence Number

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                               SPI                             |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                 64-bit Extended Sequence Number               |
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 4. AAD Format with 64-bit Extended Sequence Number

6. Packet Expansion

The initialization vector (IV) and the integrity check value (ICV) are the only sources of packet expansion. The IV always adds 8 octets to the front of the payload. The ICV is added at the end of the payload, and the CCM parameter M determines the size of the ICV. Implementations MUST support M values of 8 octets and 16 octets, and implementations MAY also support an M value of 12 octets.

7. IKE Conventions

This section describes the conventions used to generate keying material and salt values for use with AES CCM using the Internet Key Exchange (IKE) [IKE] protocol. The identifiers and attributes needed to negotiate a security association that uses AES CCM are also defined.

7.1. Keying Material and Salt Values

As previously described, implementations MUST use fresh keys with AES CCM. IKE can be used to establish fresh keys. This section describes the conventions for obtaining the unpredictable salt value for use in the nonce from IKE. Note that this convention provides a salt value that is secret as well as unpredictable.

IKE makes use of a pseudo-random function (PRF) to derive keying material. The PRF is used iteratively to derive keying material of arbitrary size, called KEYMAT. Keying material is extracted from the output string without regard to boundaries.

The size of KEYMAT MUST be three octets longer than is needed for the associated AES key. The keying material is used as follows:

AES CCM with a 128-bit key
The KEYMAT requested for each AES CCM key is 19 octets. The first 16 octets are the 128-bit AES key, and the remaining three octets are used as the salt value in the counter block.

AES CCM with a 192-bit key
The KEYMAT requested for each AES CCM key is 27 octets. The first 24 octets are the 192-bit AES key, and the remaining three octets are used as the salt value in the counter block.

AES CCM with a 256-bit key
The KEYMAT requested for each AES CCM key is 35 octets. The first 32 octets are the 256-bit AES key, and the remaining three octets are used as the salt value in the counter block.

7.2. Phase 1 Identifier

This document does not specify the conventions for using AES CCM for IKE Phase 1 negotiations. For AES CCM to be used in this manner, a separate specification is needed, and an Encryption Algorithm Identifier needs to be assigned.

7.3. Phase 2 Identifier

For IKE Phase 2 negotiations, IANA has assigned three ESP Transform Identifiers for AES CCM with an explicit IV:

14 for AES CCM with an 8-octet ICV;
15 for AES CCM with a 12-octet ICV; and
16 for AES CCM with a 16-octet ICV.

7.4. Key Length Attribute

Because the AES supports three key lengths, the Key Length attribute MUST be specified in the IKE Phase 2 exchange [DOI]. The Key Length attribute MUST have a value of 128, 192, or 256.

8. Test Vectors

Section 8 of [CCM] provides test vectors that will assist implementers with AES CCM mode.

9. Security Considerations

AES CCM employs counter (CTR) mode for confidentiality. If a counter value is ever used for more that one packet with the same key, then the same key stream will be used to encrypt both packets, and the confidentiality guarantees are voided.

What happens if the encryptor XORs the same key stream with two different packet plaintexts? Suppose two packets are defined by two plaintext byte sequences P1, P2, P3 and Q1, Q2, Q3, then both are encrypted with key stream K1, K2, K3. The two corresponding ciphertexts are:

(P1 XOR K1), (P2 XOR K2), (P3 XOR K3)

(Q1 XOR K1), (Q2 XOR K2), (Q3 XOR K3)

If both of these two ciphertext streams are exposed to an attacker, then a catastrophic failure of confidentiality results, because:

```
(P1 XOR K1) XOR (Q1 XOR K1) = P1 XOR Q1
(P2 XOR K2) XOR (Q2 XOR K2) = P2 XOR Q2
(P3 XOR K3) XOR (Q3 XOR K3) = P3 XOR Q3
```

Once the attacker obtains the two plaintexts XORed together, it is relatively straightforward to separate them. Thus, using any stream cipher, including AES CTR, to encrypt two plaintexts under the same key stream leaks the plaintext.

Therefore, AES CCM should not be used with statically configured keys. Extraordinary measures would be needed to prevent the reuse of a counter block value with the static key across power cycles. To be safe, implementations MUST use fresh keys with AES CCM. The IKE [IKE] protocol can be used to establish fresh keys.

When IKE is used to establish fresh keys between two peer entities, separate keys are established for the two traffic flows. If a different mechanism is used to establish fresh keys, one that establishes only a single key to encrypt packets, then there is a high probability that the peers will select the same IV values for some packets. Thus, to avoid counter block collisions, ESP implementations that permit use of the same key for encrypting and decrypting packets with the same peer MUST ensure that the two peers assign different salt values to the security association (SA).

Regardless of the mode used, AES with a 128-bit key is vulnerable to the birthday attack after 2^64 blocks are encrypted with a single key. Since ESP with Extended Sequence Numbers allows for up to 2^64 packets in a single SA, there is real potential for more than 2^64 blocks to be encrypted with one key. Implementations SHOULD generate a fresh key before 2^64 blocks are encrypted with the same key, or implementations SHOULD make use of the longer AES key sizes. Note that ESP with 32-bit Sequence Numbers will not exceed 2^64 blocks even if all of the packets are maximum-length Jumbograms.

10. Design Rationale

In the development of this specification, the use of the ESP sequence number field instead of an explicit IV field was considered. This section documents the rationale for the selection of an explicit IV. This selection is not a cryptographic security issue, as either approach will prevent counter block collisions.

The use of the explicit IV does not dictate the manner that the encryptor uses to assign the per-packet value in the counter block. This is desirable for several reasons.

1. Only the encryptor can ensure that the value is not used for more than one packet, so there is no advantage to selecting a mechanism that allows the decryptor to determine whether counter block values collide. Damage from the collision is done, whether the decryptor detects it or not.

2. The use of explicit IVs allows adders, LFSRs, and any other technique that meets the time budget of the encryptor, so long as the technique results in a unique value for each packet. Adders are simple and straightforward to implement, but due to carries, they do not execute in constant time. LFSRs offer an alternative that executes in constant time.

3. Complexity is in control of the implementer. Furthermore, the decision made by the implementer of the encryptor does not make the decryptor more (or less) complex.

4. The assignment of the per-packet counter block value needs to be inside the assurance boundary. Some implementations assign the sequence number inside the assurance boundary, but others do not. A sequence number collision does not have the dire consequences, but, as described in Section 6, a collision in counter block values has disastrous consequences.

5. Using the sequence number as the IV is possible in those architectures where the sequence number assignment is performed within the assurance boundary. In this situation, the sequence number and the IV field will contain the same value.

6. By decoupling the IV and the sequence number, architectures where the sequence number assignment is performed outside the assurance boundary are accommodated.

The use of an explicit IV field directly follows from the decoupling of the sequence number and the per-packet counter block value. The additional overhead (64 bits for the IV field) is acceptable. This overhead is significantly less overhead associated with Cipher Block Chaining (CBC) mode. As normally employed, CBC requires a full block for the IV and, on average, half of a block for padding. AES CCM confidentiality processing with an explicit IV has about one-third of the overhead as AES CBC, and the overhead is constant for each packet.

11. IANA Considerations

IANA has assigned three ESP transform numbers for use with AES CCM with an explicit IV:

14 for AES CCM with an 8-octet ICV;
15 for AES CCM with a 12-octet ICV; and
16 for AES CCM with a 16-octet ICV.

12. Acknowledgements

Doug Whiting and Niels Ferguson worked with me to develop CCM mode. We developed CCM mode as part of the IEEE 802.11i security effort. One of the most attractive aspects of CCM mode is that it is unencumbered by patents. I acknowledge the companies that supported the development of an unencumbered authenticated encryption mode (in alphabetical order):

Hifn
Intersil
MacFergus
RSA Security

Also, I thank Tero Kivinen for his comprehensive review of this document.

13. References

This section provides normative and informative references.

13.1. Normative References

[AES] NIST, FIPS PUB 197, "Advanced Encryption Standard (AES)," November 2001.

[DOI] Piper, D., "The Internet IP Security Domain of Interpretation for ISAKMP", RFC 2407, November 1998.

[ESP] Kent, S., "IP Encapsulating Security Payload (ESP)", RFC 4303, December 2005.

[CCM] Whiting, D., Housley, R., and N. Ferguson, "Counter with CBC-MAC (CCM)", RFC 3610, September 2003.

[STDWORDS] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

13.2. Informative References

[ARCH]	Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, December 2005.
[IKE]	Harkins, D. and D. Carrel, "The Internet Key Exchange (IKE)", RFC 2409, November 1998.
[IKEv2]	Kaufman, C., "Internet Key Exchange (IKEv2) Protocol", RFC 4306, December 2005.
[ROAD]	Thayer, R., Doraswamy, N., and R. Glenn, "IP Security Document Roadmap", RFC 2411, November 1998.
[JUMBO]	Borman, D., Deering, S., and R. Hinden, "IPv6 Jumbograms", RFC 2675, August 1999.

Author’s Address

Russell Housley
Vigil Security, LLC
918 Spring Knoll Drive
Herndon, VA 20170
USA

EMail: housley@vigilsec.com

Full Copyright Statement

Intellectual Property

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.