Network Working Group
Request for Comments: 4106
Category: Standards Track

J. Viega
Secure Software, Inc.
D. McGrew
Cisco Systems, Inc.

June 2005

# The Use of Galois/Counter Mode (GCM) in IPsec Encapsulating Security Payload (ESP)

## Status of This Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

## Copyright Notice

## Abstract

This memo describes the use of the Advanced Encryption Standard (AES) in Galois/Counter Mode (GCM) as an IPsec Encapsulating Security Payload (ESP) mechanism to provide confidentiality and data origin authentication. This method can be efficiently implemented in hardware for speeds of 10 gigabits per second and above, and is also well-suited to software implementations.

## Table of Contents

## 1. Introduction

This document describes the use of AES in GCM mode (AES-GCM) as an IPsec ESP mechanism for confidentiality and data origin authentication. We refer to this method as AES-GCM-ESP. This mechanism is not only efficient and secure, but it also enables high-speed implementations in hardware. Thus, AES-GCM-ESP allows IPsec connections that can make effective use of emerging 10-gigabit and 40-gigabit network devices.

Counter mode (CTR) has emerged as the preferred encryption method for high-speed implementations. Unlike conventional encryption modes such as Cipher Block Chaining (CBC) and Cipher Block Chaining Message Authentication Code (CBC-MAC), CTR can be efficiently implemented at high data rates because it can be pipelined. The ESP CTR protocol describes how this mode can be used with IPsec ESP [RFC3686].

Unfortunately, CTR provides no data origin authentication, and thus the ESP CTR standard requires the use of a data origin authentication algorithm in conjunction with CTR. This requirement is problematic, because none of the standard data origin authentication algorithms can be efficiently implemented for high data rates. GCM solves this problem, because under the hood, it combines CTR mode with a secure, parallelizable, and efficient authentication mechanism.

This document does not cover implementation details of GCM. Those details can be found in [GCM], along with test vectors.

### 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 [RFC2119].

## 2. AES-GCM

GCM is a block cipher mode of operation providing both confidentiality and data origin authentication. The GCM authenticated encryption operation has four inputs: a secret key, an initialization vector (IV), a plaintext, and an input for additional authenticated data (AAD). It has two outputs, a ciphertext whose length is identical to the plaintext, and an authentication tag. In the following, we describe how the IV, plaintext, and AAD are formed from the ESP fields, and how the ESP packet is formed from the ciphertext and authentication tag.

ESP also defines an IV. For clarity, we refer to the AES-GCM IV as a nonce in the context of AES-GCM-ESP. The same nonce and key combination MUST NOT be used more than once.

Because reusing an nonce/key combination destroys the security guarantees of AES-GCM mode, it can be difficult to use this mode securely when using statically configured keys. For safety's sake, implementations MUST use an automated key management system, such as the Internet Key Exchange (IKE) [RFC2409], to ensure that this requirement is met.

## 3. ESP Payload Data

The ESP Payload Data is comprised of an eight-octet initialization vector (IV), followed by the ciphertext. The payload field, as defined in [RFC2406], is structured as shown in Figure 1, along with the ICV associated with the payload.

```
    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)                          |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                                                               |
   ~                     Ciphertext (variable)                     ~
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 1: ESP Payload Encrypted with AES-GCM.

### 3.1. Initialization Vector (IV)

The AES-GCM-ESP IV field MUST be eight octets. For a given key, the IV MUST NOT repeat. The most natural way to implement this is with a counter, but anything that guarantees uniqueness can be used, such as a linear feedback shift register (LFSR). Note that the encrypter can use any IV generation method that meets the uniqueness requirement, without coordinating with the decrypter.

### 3.2.  Ciphertext

The plaintext input to AES-GCM is formed by concatenating the plaintext data described by the Next Header field with the Padding, the Pad Length, and the Next Header field. The Ciphertext field consists of the ciphertext output from the AES-GCM algorithm. The length of the ciphertext is identical to that of the plaintext.

Implementations that do not seek to hide the length of the plaintext SHOULD use the minimum amount of padding required, which will be less than four octets.

## 4.  Nonce Format

The nonce passed to the GCM-AES encryption algorithm has the following layout:

```
 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 a four-octet value that is assigned at the beginning of the security association, and then remains constant for the life of the security association. The salt SHOULD be unpredictable (i.e., chosen at random) before it is selected, but need not be secret. We describe how to set the salt for a Security Association established via the Internet Key Exchange in Section 8.1.

Initialization Vector

The IV field is described in Section 3.1.

## 5. AAD Construction

The authentication of data integrity and data origin for the SPI and (Extended) Sequence Number fields is provided without encryption. This is done by including those fields in the AES-GCM Additional Authenticated Data (AAD) field. Two formats of the AAD 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.

```
 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. Integrity Check Value (ICV)

The ICV consists solely of the AES-GCM Authentication Tag. Implementations MUST support a full-length 16-octet ICV, and MAY support 8 or 12 octet ICVs, and MUST NOT support other ICV lengths. Although ESP does not require that an ICV be present, AES-GCM-ESP intentionally does not allow a zero-length ICV. This is because GCM provides no integrity protection whatsoever when used with a zero-length Authentication Tag.

## 7. Packet Expansion

The IV adds an additional eight octets to the packet, and the ICV adds an additional 8, 12, or 16 octets. These are the only sources of packet expansion, other than the 10-13 octets taken up by the ESP SPI, Sequence Number, Padding, Pad Length, and Next Header fields (if the minimal amount of padding is used).

## 8. IKE Conventions

This section describes the conventions used to generate keying material and salt values, for use with AES-GCM-ESP, using the Internet Key Exchange (IKE) [RFC2409] protocol. The identifiers and attributes needed to negotiate a security association using AES-GCM-ESP are also defined.

### 8.1. Keying Material and Salt Values

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 the KEYMAT for the AES-GCM-ESP MUST be four octets longer than is needed for the associated AES key. The keying material is used as follows:

AES-GCM-ESP with a 128 bit key
The KEYMAT requested for each AES-GCM key is 20 octets. The first 16 octets are the 128-bit AES key, and the remaining four octets are used as the salt value in the nonce.

AES-GCM-ESP with a 192 bit key
The KEYMAT requested for each AES-GCM key is 28 octets. The first 24 octets are the 192-bit AES key, and the remaining four octets are used as the salt value in the nonce.

AES-GCM-ESP with a 256 bit key
The KEYMAT requested for each AES GCM key is 36 octets. The first 32 octets are the 256-bit AES key, and the remaining four octets are used as the salt value in the nonce.

### 8.2. Phase 1 Identifier

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

Phase 1 cipher that is at least as strong as AES-GCM. The use of AES CBC [RFC3602] with the same key size used by AES-GCM-ESP is RECOMMENDED.

## 8.3. Phase 2 Identifier

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

18 for AES-GCM with an 8 octet ICV;
19 for AES-GCM with a 12 octet ICV; and
20 for AES-GCM with a 16 octet ICV.

## 8.4. Key Length Attribute

Because the AES supports three key lengths, the Key Length attribute MUST be specified in the IKE Phase 2 exchange [RFC2407]. The Key Length attribute MUST have a value of 128, 192, or 256.

## 9. Test Vectors

Appendix B of [GCM] provides test vectors that will assist implementers with AES-GCM mode.

## 10. Security Considerations

GCM is provably secure against adversaries that can adaptively choose plaintexts, ciphertexts, ICVs, and the AAD field, under standard cryptographic assumptions (roughly, that the output of the underlying cipher, under a randomly chosen key, is indistinguishable from a randomly selected output). Essentially, this means that, if used within its intended parameters, a break of GCM implies a break of the underlying block cipher. The proof of security for GCM is available in [GCM].

The most important security consideration is that the IV never repeat for a given key. In part, this is handled by disallowing the use of AES-GCM when using statically configured keys, as discussed in Section 2.

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).

The other consideration is that, as with any encryption mode, the security of all data protected under a given security association decreases slightly with each message.

To protect against this problem, implementations MUST generate a fresh key before encrypting 2^64 blocks of data with a given key. Note that it is impossible to reach this limit when using 32-bit Sequence Numbers.

Note that, for each message, GCM calls the block cipher once for each full 16-octet block in the payload, once for any remaining octets in the payload, and one additional time for computing the ICV.

Clearly, smaller ICV values are more likely to be subject to forgery attacks. Implementations SHOULD use as large a size as reasonable.

## 11. Design Rationale

This specification was designed to be as similar to the AES-CCM ESP [CCM-ESP] and AES-CTR ESP [RFC3686] mechanisms as reasonable, while promoting simple, efficient implementations in both hardware and software. We re-use the design and implementation experience from those standards.

The major difference with CCM is that the CCM ESP mechanism requires an 11-octet nonce, whereas the GCM ESP mechanism requires using a 12-octet nonce. GCM is specially optimized to handle the 12-octet nonce case efficiently. Nonces of other lengths would cause unnecessary, additional complexity and delays, particularly in hardware implementations. The additional octet of nonce is used to increase the size of the salt.

## 12. IANA Considerations

IANA has assigned three ESP Transform Identifiers for AES-GCM with an eight-byte explicit IV:

18 for AES-GCM with an 8 octet ICV;
19 for AES-GCM with a 12 octet ICV; and
20 for AES-GCM with a 16 octet ICV.

## 13. Acknowledgements

This work is closely modeled after Russ Housley’s AES-CCM transform [CCM-ESP]. Portions of this document are directly copied from that work in progress. We thank Russ for his support of this work.

Additionally, the GCM mode of operation was originally conceived as an improvement to Carter-Wegman Counter (CWC) mode [CWC], the first unencumbered block cipher mode capable of supporting high-speed authenticated encryption.

## 14. Normative References

[GCM] McGrew, D. and J. Viega, "The Galois/Counter Mode of Operation (GCM)", Submission to NIST. http://csrc.nist.gov/CryptoToolkit/modes/proposedmodes/gcm/gcm-spec.pdf, January 2004.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2406] Kent, S. and R. Atkinson, "IP Encapsulating Security Payload (ESP)", RFC 2406, November 1998.

[RFC2407] Piper, D., "The Internet IP Security Domain of Interpretation for ISAKMP", RFC 2407, November 1998.

[RFC3602] Frankel, S., Glenn, R. and S. Kelly, "The AES-CBC Cipher Algorithm and Its Use with IPsec", RFC 3602, September 2003.

## 15. Informative References

[CCM-ESP] Housley, R., "Using AES CCM Mode With IPsec ESP", Work In Progress.

[CWC] Kohno, T., Viega, J. and D. Whiting, "CWC: A high-performance conventional authenticated encryption mode", Fast Software Encryption. http://eprint.iacr.org/2003/106.pdf, February 2004.

[RFC2409] Harkins, D. and D. Carrel, "The Internet Key Exchange (IKE)", RFC 2409, November 1998.

[RFC3686] Housley, R., "Using Advanced Encryption Standard (AES) Counter Mode With IPsec Encapsulating Security Payload (ESP)", RFC 3686, January 2004.

## Authors' Addresses

John Viega
Secure Software, Inc.
4100 Lafayette Center Dr., Suite 100
Chantilly, VA  20151
US

Phone: (703) 814 4402
EMail: viega@securesoftware.com

David A. McGrew
Cisco Systems, Inc.
510 McCarthy Blvd.
Milpitas, CA  95035
US

Phone: (408) 525 8651
EMail: mcgrew@cisco.com
URI:   http://www.mindspring.com/~dmcgrew/dam.htm

## Full Copyright Statement

## Intellectual Property

## Acknowledgement

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