

Network Working Group  
Internet Draft  
Intended Status: Informational  
Expires: November 21, 2009

K.M. Igoe  
National Security Agency  
May 20, 2009  
J.A. Solinas  
National Security Agency  
May 20, 2009

AES Galois Counter Mode for the Secure Shell Transport Layer Protocol  
draft-igoe-secsh-aes-gcm-02

#### Status of this Memo

This Internet-Draft is submitted to IETF in full conformance with the provisions of [BCP 78](#) and [BCP 79](#).

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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."

The list of current Internet-Drafts can be accessed at <http://www.ietf.org/ietf/lid-abstracts.txt>.

The list of Internet-Draft Shadow Directories can be accessed at <http://www.ietf.org/shadow.html>.

This Internet-Draft will expire on November 21, 2009.

#### Copyright Notice

Copyright (c) 2009 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to [BCP 78](#) and the IETF Trust's Legal Provisions Relating to IETF Documents in effect on the date of publication of this document (<http://trustee.ietf.org/license-info>). Please review these documents carefully, as they describe your rights and restrictions with respect to this document.

Internet Draft

AES-GCM for Secure Shell

May 20, 2009

## Abstract

Secure Shell (SSH, [RFC 4251](#)) is a secure remote-login protocol. SSH provides for algorithms that provide authentication, key agreement, confidentiality and data integrity services. The purpose of this document is to show how the AES Galois/Counter Mode can be used to provide both confidentiality and data integrity to the SSH Transport Layer

## Table of Contents

<a href="#">1.</a>	Introduction.....	<a href="#">2</a>
<a href="#">2.</a>	Requirements Terminology.....	<a href="#">2</a>
<a href="#">3.</a>	Applicability Statement.....	<a href="#">2</a>
<a href="#">4.</a>	Properties of Galois Counter Mode.....	<a href="#">3</a>
<a href="#">4.1.</a>	AES GCM Authenticated Encryption.....	<a href="#">3</a>
<a href="#">4.2.</a>	AES GCM Authenticated Decryption.....	<a href="#">3</a>
<a href="#">5.</a>	Review of Secure Shell.....	<a href="#">4</a>
<a href="#">5.1.</a>	Key Exchange.....	<a href="#">4</a>
<a href="#">5.2.</a>	Secure Shell Binary Packets.....	<a href="#">5</a>
<a href="#">5.2.1.</a>	Treatment of the Packet Length Field.....	<a href="#">5</a>
<a href="#">6.</a>	Two New AEAD Algorithms.....	<a href="#">6</a>
<a href="#">6.1.</a>	aead-aes-128-gcm-ssh.....	<a href="#">6</a>
<a href="#">6.2.</a>	aead-aes-256-gcm-ssh.....	<a href="#">6</a>
<a href="#">7.</a>	IV and Counter Management.....	<a href="#">7</a>
<a href="#">8.</a>	Size of the Message Authentication Code.....	<a href="#">7</a>
<a href="#">9.</a>	Security Considerations.....	<a href="#">7</a>
<a href="#">9.1.</a>	Use of Packet Sequence Number in MAC.....	<a href="#">8</a>
<a href="#">9.2.</a>	Non-encryption of Packet Length.....	<a href="#">8</a>
<a href="#">10.</a>	IANA Considerations.....	<a href="#">9</a>
<a href="#">11.</a>	References.....	<a href="#">9</a>
<a href="#">11.1.</a>	Normative References.....	<a href="#">9</a>

[1.](#) Introduction

Galois/Counter Mode (GCM) is a block cipher mode of operation that provides both confidentiality and data integrity services. The purpose of this document is to show how AES-GCM can be integrated

into the Secure Shell Transport Layer Protocol, [RFC 4253](#).

## [2. Requirements Terminology](#)

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](#)].

## [3. Applicability Statement](#)

Igoe and Solinas

Informational

[Page 2]

---

Internet Draft

AES-GCM for Secure Shell

May 20, 2009

Using AES-GCM to provide both confidentiality and data integrity is generally more efficient than using two separate algorithms to provide these security services.

## [4. Properties of Galois Counter Mode](#)

Galois Counter Mode (GCM) is a mode of operation for block ciphers which provides both confidentiality and data integrity. NIST Special Publication SP 800 38D [[GCM](#)] gives an excellent explanation of Galois Counter Mode. In this document we shall focus on AES GCM, the use of the Advanced Encryption Algorithm (AES) in Galois Counter Mode. AES-GCM is an example of an "algorithm for authenticated encryption with associated data" (AEAD algorithm) as described in [[RFC5116](#)].

### [4.1. AES GCM Authenticated Encryption](#)

An invocation of AES GCM to perform an authenticated encryption has the following inputs and outputs:

GCM Authenticated Encryption

Inputs:

```
    octet_string PT ;    // Plain text, to be both
                        //   authenticated and encrypted
    octet_string AAD;    // Additional Authenticated Data,
                        //   authenticated but not encrypted
    octet_string IV;    // Initialization vector
```

```
octet_string BK; // Block cipher key
```

Outputs:

```
octet_string CT; // Cipher Text
octet_string AT; // Authentication Tag
```

Note: in [\[RFC5116\]](#) the IV is called the nonce.

For a given block cipher key BK it is critical that no IV be used more than once. [Section 6](#) addresses how this goal is to be achieved in secure shell.

#### [4.2](#). AES GCM Authenticated Decryption

An invocation of AES GCM to perform an authenticated decryption has the following inputs and outputs:

GCM Authenticated Decryption

Inputs:

Igoe and Solinas

Informational

[Page 3]

---

Internet Draft

AES-GCM for Secure Shell

May 20, 2009

```
octet_string CT ; // Cipher text, to be both
                  //   authenticated and decrypted.
octet_string AAD; // Additional Authenticated Data,
                  //   authenticated only.
octet_string AT; // Authentication Tag
octet_string IV; // Initialization vector
octet_string BK; // Block cipher key.
```

Output:

```
Failure_Indicator; // Returned if the authentication tag
                   //   is invalid.
octet_string PT; // Plain text, returned if and only if
                 //   the authentication tag is valid.
```

AES-GCM is prohibited from returning any portion of the plaintext until the authentication tag has been validated. Though this feature greatly simplifies the security analysis of any system using AES-GCM, as we shall see in [section 5.2.1](#), this creates an incompatibility with the requirements of secure shell.

## 5. Review of Secure Shell

The goal of secure shell is to establish two secure tunnels between a client and a server, one tunnel carrying client-to-server communications and the other server-to-client communications. Each tunnel is encrypted and a message authentication code is used to insure data integrity.

### 5.1. Key Exchange

These tunnels are initialized using the secure shell key exchange protocol as described in [section 7 of \[RFC4253\]](#). This protocol negotiates a mutually acceptable set of cryptographic algorithms, and produces a secret value K and an exchange hash H shared by the client and server. The initial value of H is saved for use as the session\_id.

If AES-GCM is selected as the encryption algorithm for a given tunnel, AES-GCM MUST also be selected as the mac algorithm. Conversely, if AES-GCM is selected as the mac algorithm, it MUST also be selected as the encryption algorithm.

As described in [section 7.2 of \[RFC4253\]](#), a hash based key derivation function (KDF) is applied to the shared secret value K to generate the required symmetric keys. Each tunnel gets a distinct set of symmetric keys. The keys are generated as shown in figure 1. The sizes of these keys varies depending upon which cryptographic algorithms are being used.

#### Initial IV

Client-to-Sever      HASH( K || H ||"A"|| session\_id)

Server-to-Client    HASH( K || H ||"B"|| session\_id)

#### Encryption Key

Client-to-Sever      HASH( K || H ||"C"|| session\_id)

Server-to-Client    HASH( K || H ||"D"|| session\_id)

#### Integrity Key

Client-to-Sever      HASH( K || H ||"E"|| session\_id)

Server-to-Client    HASH( K || H ||"F"|| session\_id)

## Figure 1: Key Derivation in Secure Shell

As we shall see below, SSH AES-GCM requires a 12-octet Initial IV and an encryption key of either 16 or 32 octets. Because an AEAD algorithm such as AES-GCM uses the encryption key to provide both confidentiality and data integrity, the integrity key is not used with AES-GCM.

Either the server or client may at any time request that the secure shell session be rekeyed. The shared secret value K, the exchange hash H, and all the above symmetric keys will be updated. Only the session\_id will remain unchanged.

### 5.2. Secure Shell Binary Packets

Upon completion of the key exchange protocol, all further secure shell traffic is parsed into a data structure known as a secure shell binary packet as shown below in Figure 2 (see also [section 6 of \[RFC4253\]](#)).

```
uint32    packet_length; // 0 <= packet_length < 2^32
byte      padding_length; // 4 <= padding_length < 256
byte[n1]  payload;        // n1 = packet_length-padding_length-1
byte[n2]  random_padding; // n2 = padding_length
byte[m]   mac;            // m = mac_length
```

Figure 2: Structure of a Secure Shell Binary Packet

The authentication tag produced by AES-GCM authenticated encryption will be placed in the mac field at the end of the secure shell binary packet.

#### 5.2.1. Treatment of the Packet Length Field

[Section 6.3 of \[RFC4253\]](#) requires that the packet length, padding length, payload and padding fields of each binary packet be encrypted. This presents a problem for SSH AES-GCM because:

- 1) The tag can not be verified until we parse the binary packet
- 2) The packet can not be parsed until the packet\_length has been

- decrypted.
- 3) The `packet_length` can not be decrypted until the tag has been verified.

When using AES-GCM with secure shell, the `packet_length` field is to be treated as additional authenticated data, not as plaintext. This violates the requirements of [\[RFC4253\]](#). The repercussions of this decision are discussed in the security considerations section of this document.

## [6.](#) Two New AEAD Algorithms

### [6.1.](#) `aead-aes-128-gcm-ssh`

`aead-aes-128-gcm-ssh` is a variant of the algorithm `AEAD_AES_128_GCM` specified in [section 5.1 of \[RFC5116\]](#). The only differences between the two algorithms are in the input and output lengths. Using the notation defined in [\[RFC5116\]](#), the input and output lengths for `aead-aes-128-gcm-ssh` are as follows:

PARAMETER	Meaning	Value
<code>K_LEN</code>	AES key length	16 octets
<code>P_MAX</code>	maximum plaintext length	$2^{32} - 32$ octets
<code>A_MAX</code>	maximum additional authenticated data length	4 octets
<code>N_MIN</code>	minimum nonce (IV) length	12 octets
<code>N_MAX</code>	maximum nonce (IV) length	12 octets
<code>C_MAX</code>	maximum cipher length	$2^{32} - 32$ octets

Test cases are provided in the appendix of [\[GCM\]](#).

The reader is reminded that due to the presence of length fields and padding in SSH packets, the plaintext length is not the same as the payload length. See [section 4.2](#) above.

### [6.2.](#) `aead-aes-256-gcm-ssh`

`aead-aes-256-gcm-ssh` is a variant of the algorithm `AEAD_AES_256_GCM` specified in [section 5.2 of \[RFC5116\]](#). The only differences between the two algorithms are in the input and output lengths. Using the notation defined in [\[RFC5116\]](#), the input and output lengths for `aead-aes-256-gcm-ssh` are as follows:

PARAMETER	Meaning	Value
<code>K_LEN</code>	AES key length	32 octets
<code>P_MAX</code>	maximum plaintext length	$2^{32} - 32$ octets
<code>A_MAX</code>	maximum additional	4 octets

Internet Draft

AES-GCM for Secure Shell

May 20, 2009

N_MIN	minimum nonce (IV) length	12 octets
N_MAX	maximum nonce (IV) length	12 octets
C_MAX	maximum cipher length	$2^{32} - 32$ octets

Test cases are provided in the appendix of [\[GCM\]](#).

The reader is reminded that due to the presence of length fields and padding in SSH packets, the plaintext length is not the same as the payload length. See [section 4.2](#) above.

## 7. IV and Counter Management

With AES-GCM, the 12-octet IV is broken into two fields: a 4-octet fixed field and an 8-octet invocation counter field. The invocation field is treated as a 64-bit integer and is incremented after each invocation of AES-GCM to process a binary packet.

```
uint32  fixed;           // 4 octets
uint64  invocation_counter; // 8 octets
```

Figure 3: Structure of an SSH AES-GCM nonce

AES-GCM produces a keystream in blocks of 16-octets which is used to encrypt the plaintext. This keystream is produced by encrypting the following 16-octet data structure:

```
uint32  fixed;           // 4 octets
uint64  invocation_counter; // 8 octets
uint32  block_counter;   // 4 octets
```

Figure 4: Structure of an AES input for SSH AES-GCM

The `block_counter` is initially set to one (1) and incremented as each block of key is produced.

The reader is reminded that SSH requires that the data to be encrypted MUST be padded out to a multiple of the block size (16-octets for AES-GCM).



## [8.](#) Size of the Message Authentication Code

Both `aead-aes-128-gcm-ssh` and `aead-aes-256-gcm-ssh` produce a 16-octet message authentication code. ([\[RFC5116\]](#) calls this an "authentication tag" rather than a "message authentication code".)

## [9.](#) Security Considerations

The security considerations in [\[RFC4251\]](#) apply.

### [9.1.](#) Use of Packet Sequence Number in MAC

[\[RFC4253\]](#) requires that the formation of the mac involve the `packet_sequence_number`, a 32-bit value that counts the number of binary packets that have been sent on a given SSH tunnel. Since the `sequence_number` is, up to an additive constant, just the low 32-bits of the `invocation_counter`, the presence of the `invocation_counter` field in the IV insures that the `sequence_number` is indeed involved in the formation of the integrity tag, though this involvement differs slightly from the requirements in [section 6.4 of \[RFC4253\]](#).

### [9.2.](#) Non-encryption of Packet Length

As discussed in [section 5.2.1](#), there is an incompatibility between GCM's requirement that no plaintext be returned until the authentication tag has been verified, secure shell's requirement that the packet length be encrypted, and the necessity of decrypting the packet length field to locate the authentication tag. This document addresses this dilemma by requiring that, in AES-GCM, the packet length field will not be encrypted but will instead be processed as Additional Authenticated Data.

In theory, one could argue that encryption of the entire binary packet means that the secure shell dataflow becomes a featureless octet stream. But in practice, the secure shell dataflow will come in bursts, with the length of each burst strongly correlated to the length of the underlying binary packets. Encryption of the packet length does little in and of itself to disguise the length of the underlying binary packets. Secure shell provides two other

mechanisms, random padding and SSH\_MSG\_IGNORE messages, that are far more effective than encrypting the packet length in masking any structure in the underlying plaintext stream that might be revealed by the length of the binary packets.

## 10. IANA Considerations

IANA will add the following two entries to the AEAD Registry described in [[RFC5116](#)]:

Name	Reference	Proposed Numeric Identifier
aead-aes-128-gcm-ssh	<a href="#">Section 5.1</a>	5
aead-aes-256-gcm-ssh	<a href="#">Section 5.2</a>	6

IANA will add the following two entries to the Secure Shell Encryption Algorithm name Registry described in [[RFC4250](#)]:

Name	Reference
------	-----------

aead-aes-128-gcm-ssh	<a href="#">Section 5.1</a>
aead-aes-256-gcm-ssh	<a href="#">Section 5.2</a>

IANA will add the following two entries to the Secure Shell MAC Algorithm name Registry described in [RF4250]:

Name	Reference
aead-aes-128-gcm-ssh	<a href="#">Section 5.1</a>
aead-aes-256-gcm-ssh	<a href="#">Section 5.2</a>

## [11](#). References

### [11.1](#). Normative References

- [GCM] Dworkin, M, "Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC", NIST Special Publication 800-38D, November 2007.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", [BCP 14](#), [RFC 2119](#), March 1997.

- [RFC4250] Lehtinen, S. and C. Lonvick, Ed., "The Secure Shell (SSH) Protocol Assigned Numbers", [RFC 4250](#), January 2006.
- [RFC4251] Ylonen, T. and C. Lonvick, Ed., "The Secure Shell (SSH) Protocol Architecture", [RFC 4251](#), January 2006.
- [RFC4253] Ylonen, T. and C. Lonvick, Ed., "The Secure Shell (SSH) Transport Layer Protocol", [RFC 4253](#), January 2006
- [RFC4344] Bellare, M., Kohno, T., and C. Namprempre, "The Secure

Shell (SSH) Transport Layer Encryption Modes", [RFC 4344](#),  
January 2006.

[RFC5116] McGrew, D., "An Interface and Algorithms for Authenticated  
Encryptions", [RFC 5116](#), January 2008.

Kevin M. Igoe  
NSA/CSS Commercial Solutions Center  
National Security Agency  
EMail: kmigoe@nsa.gov

Jerome A. Solinas  
National Information Assurance Research Laboratory  
National Security Agency  
EMail: jasolin@orion.ncsc.mil

#### Acknowledgement

Funding for the RFC Editor function is provided by the IETF  
Administrative Support Activity (IASA).

