Network Working Group
Request for Comments: 3566
Category: Standards Track

S. Frankel
NIST
H. Herbert
Intel

September 2003

The AES-XCBC-MAC-96 Algorithm and Its Use With IPsec

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

A Message Authentication Code (MAC) is a key-dependent one way hash function. One popular way to construct a MAC algorithm is to use a block cipher in conjunction with the Cipher-Block-Chaining (CBC) mode of operation. The classic CBC-MAC algorithm, while secure for messages of a pre-selected fixed length, has been shown to be insecure across messages of varying lengths such as the type found in typical IP datagrams. This memo specifies the use of AES in CBC mode with a set of extensions to overcome this limitation. This new algorithm is named AES-XCBC-MAC-96.

Table of Contents

1. Introduction

Message authentication provides data integrity and data origin authentication with respect to the original message source. A Message Authentication Code (MAC) is a key-dependent one way hash function. One popular way to construct a MAC algorithm is to use a block cipher in conjunction with the Cipher-Block-Chaining (CBC) mode of operation. The classic CBC-MAC algorithm, while secure for messages of a pre-selected fixed length [CBC-MAC-2], has been shown to be insecure across messages of varying lengths such as the type found in typical IP datagrams [CBC-MAC-2, section 5]. In fact, it is trivial to produce forgeries for a second message given the MAC of a prior message. [HANDBOOK, section 9.62, p. 354]

This memo specifies the use of AES [AES] in CBC mode [MODES] with a set of extensions [XCBC-MAC-1] to overcome this limitation. This new algorithm is named AES-XCBC-MAC-96. Using the AES block cipher, with its increased block size (128 bits) and increased key length (128 bits), provides the new algorithm with the ability to withstand continuing advances in crypto-analytic techniques and computational capability. AES-XCBC-MAC-96 is used as an authentication mechanism within the context of the IPsec Encapsulating Security Payload (ESP) and the Authentication Header (AH) protocols. For further information on ESP, refer to [ESP] and [ROADMAP]. For further information on AH, refer to [AH] and [ROADMAP].

The goal of AES-XCBC-MAC-96 is to ensure that the datagram is authentic and cannot be modified in transit. Data integrity and data origin authentication as provided by AES-XCBC-MAC-96 are dependent upon the scope of the distribution of the secret key. If the key is known only by the source and destination, this algorithm will provide both data origin authentication and data integrity for datagrams sent between the two parties. In addition, only a party with the identical key can verify the hash.

2. Specification of Requirements

The keywords "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" that appear in this document are to be interpreted as described in BCP 14, RFC 2119 [RFC-2119].

3. Basic CBC-MAC with Obligatory 10* Padding

CBC-MAC uses a block cipher for encryption; the block cipher transforms b bits of plaintext to b bits of ciphertext. The basic CBC-MAC [CBC-MAC-1, CBC-MAC-2] with Obligatory 10* Padding over a b-bit block cipher is calculated as follows for a message M:

(1) Append a single 1 bit to M. Then append the minimum number of 0 bits to M such that the length of M is a multiple of b. [NOTE: This is 1 of several padding schemes that can be used for CBC-MAC. Several others are described in [MODES].]

(2) Break M into n blocks, M[1] ... M[n], where the blocksize of blocks M[1] ... M[n] is b bits

(3) Define E[0] = 0x00000000000000000000000000000000

(4) For each block M[i], where i = 1 ... n:
XOR M[i] with E[i-1], then encrypt the result with Key K, yielding E[i].

(5) E[n] is the b-bit authenticator.

Basic CBC-MAC with obligatory 10* padding has been shown to be secure for messages up to (but not including) a pre-selected fixed length, in which the length is a multiple of the blocksize. This algorithm is not suitable for IPsec for the following reasons:

+ Any IPsec authenticator must be able to handle messages of arbitrary length. However, the basic CBC-MAC cannot securely handle messages that exceed the pre-selected fixed length.

+ For messages shorter than the pre-selected fixed length, padding the message to the pre-selected fixed length may necessitate additional encryption operations, adding an unacceptable computational penalty.

4. AES-XCBC-MAC-96

[AES] describes the underlying AES algorithm, while [CBC-MAC-1] and [XCBC-MAC-1] describe the AES-XCBC-MAC algorithm.

The AES-XCBC-MAC-96 algorithm is a variant of the basic CBC-MAC with obligatory 10* padding; however, AES-XCBC-MAC-96 is secure for messages of arbitrary length. The AES-XCBC-MAC-96 calculations require numerous encryption operations; this encryption MUST be accomplished using AES with a 128-bit key. Given a 128-bit secret key K, AES-XCBC-MAC-96 is calculated as follows for a message M that

consists of n blocks, M[1] ... M[n], in which the blocksize of blocks M[1] ... M[n-1] is 128 bits and the blocksize of block M[n] is between 1 and 128 bits:

(1) Derive 3 128-bit keys (K1, K2 and K3) from the 128-bit secret key K, as follows:
K1 = 0x01010101010101010101010101010101 encrypted with Key K
K2 = 0x02020202020202020202020202020202 encrypted with Key K
K3 = 0x03030303030303030303030303030303 encrypted with Key K

(2) Define E[0] = 0x00000000000000000000000000000000

(3) For each block M[i], where i = 1 ... n-1:
XOR M[i] with E[i-1], then encrypt the result with Key K1, yielding E[i].

(4) For block M[n]:

a) If the blocksize of M[n] is 128 bits:
XOR M[n] with E[n-1] and Key K2, then encrypt the result with Key K1, yielding E[n].

b) If the blocksize of M[n] is less than 128 bits:

i) Pad M[n] with a single "1" bit, followed by the number of "0" bits (possibly none) required to increase M[n]'s blocksize to 128 bits.

ii) XOR M[n] with E[n-1] and Key K3, then encrypt the result with Key K1, yielding E[n].

(5) The authenticator value is the leftmost 96 bits of the 128-bit E[n].

NOTE1: If M is the empty string, pad and encrypt as in (4)(b) to create M[1] and E[1]. This will never be the case for ESP or AH, but is included for completeness sake.

NOTE2: [CBC-MAC-1] defines K1 as follows:

K1 = Constant1A encrypted with Key K |
Constant1B encrypted with Key K.

However, the second encryption operation is only needed for AES-XCBC-MAC with keys greater than 128 bits; thus, it is not included in the definition of AES-XCBC-MAC-96.

AES-XCBC-MAC-96 verification is performed as follows:

Upon receipt of the AES-XCBC-MAC-96 authenticator, the entire 128-bit value is computed and the first 96 bits are compared to the value stored in the authenticator field.

4.1. Keying Material

AES-XCBC-MAC-96 is a secret key algorithm. For use with either ESP or AH a fixed key length of 128-bits MUST be supported. Key lengths other than 128-bits MUST NOT be supported (i.e., only 128-bit keys are to be used by AES-XCBC-MAC-96).

AES-XCBC-MAC-96 actually requires 384 bits of keying material (128 bits for the AES keysize + 2 times the blocksize). This keying material can either be provided through the key generation mechanism or it can be generated from a single 128-bit key. The latter approach has been selected for AES-XCBC-MAC-96, since it is analogous to other authenticators used within IPsec. The reason AES-XCBC-MAC-96 uses 3 keys is so the length of the input stream does not need to be known in advance. This may be useful for systems that do one-pass assembly of large packets.

A strong pseudo-random function MUST be used to generate the required 128-bit key. This key, along with the 3 derived keys (K1, K2 and K3), should be used for no purposes other than those specified in the algorithm. In particular, they should not be used as keys in another cryptographic setting. Such abuses will invalidate the security of the authentication algorithm.

At the time of this writing there are no specified weak keys for use with AES-XCBC-MAC-96. This does not mean to imply that weak keys do not exist. If, at some point, a set of weak keys for AES-XCBC-MAC-96 are identified, the use of these weak keys MUST be rejected followed by a request for replacement keys or a newly negotiated Security Association.

[ARCH] describes the general mechanism for obtaining keying material when multiple keys are required for a single SA (e.g., when an ESP SA requires a key for confidentiality and a key for authentication).

In order to provide data origin authentication, the key distribution mechanism must ensure that unique keys are allocated and that they are distributed only to the parties participating in the communication.

Current attacks do not necessitate a specific recommended frequency for key changes. However, periodic key refreshment is a fundamental security practice that helps against potential weaknesses of the function and the keys, reduces the information available to a cryptanalyst, and limits the damage resulting from a compromised key.

4.2. Padding

AES-XCBC-MAC-96 operates on 128-bit blocks of data. Padding requirements are specified in [CBC-MAC-1] and are part of the XCBC algorithm. If you build AES-XCBC-MAC-96 according to [CBC-MAC-1] you do not need to add any additional padding as far as AES-XCBC-MAC-96 is concerned. With regard to "implicit packet padding" as defined in [AH], no implicit packet padding is required.

4.3. Truncation

AES-XCBC-MAC produces a 128-bit authenticator value. AES-XCBC-MAC-96 is derived by truncating this 128-bit value as described in [HMAC] and verified in [XCBC-MAC-2]. For use with either ESP or AH, a truncated value using the first 96 bits MUST be supported. Upon sending, the truncated value is stored within the authenticator field. Upon receipt, the entire 128-bit value is computed and the first 96 bits are compared to the value stored in the authenticator field. No other authenticator value lengths are supported by AES-XCBC-MAC-96.

The length of 96 bits was selected because it is the default authenticator length as specified in [AH] and meets the security requirements described in [XCBC-MAC-2].

4.4. Interaction with the ESP Cipher Mechanism

As of this writing, there are no known issues which preclude the use of AES-XCBC-MAC-96 with any specific cipher algorithm.

4.5. Performance

For any CBC MAC variant, the major computational effort is expended in computing the underlying block cipher. This algorithm uses a minimum number of AES invocations, one for each block of the message or fraction thereof, resulting in performance equivalent to classic CBC-MAC.

The key expansion requires 3 additional AES encryption operations, but these can be performed once in advance for each secret key.

4.6. Test Vectors

These test cases were provided by John Black, co-author of the XCBC-MAC algorithm, who verified them with 2 independent implementations. All values are hexadecimal numbers.

```
Test Case #1   : AES-XCBC-MAC-96 with 0-byte input
Key (K)        : 000102030405060708090a0b0c0d0e0f
Message (M)    : <empty string>
AES-XCBC-MAC   : 75f0251d528ac01c4573dfd584d79f29
AES-XCBC-MAC-96: 75f0251d528ac01c4573dfd5

Test Case #2   : AES-XCBC-MAC-96 with 3-byte input
Key (K)        : 000102030405060708090a0b0c0d0e0f
Message (M)    : 000102
AES-XCBC-MAC   : 5b376580ae2f19afe7219ceef172756f
AES-XCBC-MAC-96: 5b376580ae2f19afe7219cee

Test Case #3   : AES-XCBC-MAC-96 with 16-byte input
Key (K)        : 000102030405060708090a0b0c0d0e0f
Message (M)    : 000102030405060708090a0b0c0d0e0f
AES-XCBC-MAC   : d2a246fa349b68a79998a4394ff7a263
AES-XCBC-MAC-96: d2a246fa349b68a79998a439

Test Case #4   : AES-XCBC-MAC-96 with 20-byte input
Key (K)        : 000102030405060708090a0b0c0d0e0f
Message (M)    : 000102030405060708090a0b0c0d0e0f10111213
AES-XCBC-MAC   : 47f51b4564966215b8985c63055ed308
AES-XCBC-MAC-96: 47f51b4564966215b8985c63

Test Case #5   : AES-XCBC-MAC-96 with 32-byte input
Key (K)        : 000102030405060708090a0b0c0d0e0f
Message (M)    : 000102030405060708090a0b0c0d0e0f1011121314151617181
                 1a1b1c1d1e1f
AES-XCBC-MAC   : f54f0ec8d2b9f3d36807734bd5283fd4
AES-XCBC-MAC-96: f54f0ec8d2b9f3d36807734b

Test Case #6   : AES-XCBC-MAC-96 with 34-byte input
Key (K)        : 000102030405060708090a0b0c0d0e0f
Message (M)    : 000102030405060708090a0b0c0d0e0f1011121314151617181
                 1a1b1c1d1e1f2021
AES-XCBC-MAC   : becbb3bccdb518a30677d5481fb6b4d8
AES-XCBC-MAC-96: becbb3bccdb518a30677d548

Test Case #7   : AES-XCBC-MAC-96 with 1000-byte input
Key (K)        : 000102030405060708090a0b0c0d0e0f
Message (M)    : 0000000000000000000 ... 0000000000000000000
                 [1000 bytes]
```

```
AES-XCBC-MAC    : f0dafee895db30253761103b5d84528f
AES-XCBC-MAC-96: f0dafee895db30253761103b
```

5. Security Considerations

The security provided by AES-XCBC-MAC-96 is based upon the strength of AES. At the time of this writing there are no practical cryptographic attacks against AES or AES-XCBC-MAC-96.

As is true with any cryptographic algorithm, part of its strength lies in the correctness of the algorithm implementation, the security of the key management mechanism and its implementation, the strength of the associated secret key, and upon the correctness of the implementation in all of the participating systems. This document contains test vectors to assist in verifying the correctness of AES-XCBC-MAC-96 code.

6. IANA Considerations

IANA has assigned AH Transform Identifier 9 to AH_AES-XCBC-MAC. IANA has assigned AH/ESP Authentication Algorithm Value 9 to AES-XCBC-MAC.

7. Intellectual Property Rights Statement

8. Acknowledgments

Portions of this text were unabashedly borrowed from [HMAC-SHA].

Thanks to the XCBC-MAC authors for their expert advice and rapid response to our queries: to Phil Rogaway for providing values for the XCBC-MAC constants; and to John Black for detailed corrections to the algorithm specifications and for providing the test cases. Thanks also to Andrew Krywaniuk for insisting on (and providing wording for) a rationale for the 3-key approach.

9. References

9.1. Normative References

[AES] NIST, FIPS PUB 197, "Advanced Encryption Standard (AES)," November 2001. http://csrc.nist.gov/publications/fips/fips197/fips-197.{ps,pdf}

[AH] Kent, S. and R. Atkinson, "IP Authentication Header", RFC 2402, November 1998.

[CBC-MAC-1] Black, J. and P. Rogaway, "CBC MACs for Arbitrary-Length Messages: The Three-Key Constructions," in M. Bellare, editor, Advances in Cryptology -- CRYPTO '00, volume 1880 of Lecture Notes in Computer Science, p. 0197, August 2000, Springer-Verlag. http://www.cs.ucdavis.edu/~rogaway/papers/3k.ps

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

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

[XCBC-MAC-1] Black, J. and P. Rogaway, "A Suggestion for Handling Arbitrary-Length Messages with the CBC MAC," NIST Second Modes of Operation Workshop, August 2001. http://csrc.nist.gov/encryption/modes/proposedmodes/xcbc-mac/xcbc-mac-spec.pdf

9.2. Informative References

[ARCH] Kent, S. and R. Atkinson, "Security Architecture for the Internet Protocol", RFC 2401, November 1998.

[CBC-MAC-2] Bellare, M., J. Kilian and P. Rogaway, "The Security of the Cipher Block Chaining Message Authentication Code," Journal of Computer and System Sciences (JCSS), Vol. 61, No. 3, December 2000, pp. 362-399. http://www.cse.ucsd.edu/users/mihir/papers/cbc.{ps,pdf}

[HMAC] Krawczyk, H., Bellare, M. and R. Canetti, "HMAC: Keyed-Hashing for Message Authentication", RFC 2104, February 1997.

[HMAC-SHA] Madson, C. and R. Glenn, "The Use of HMAC-SHA-1-96 within ESP and AH", RFC 2404, November 1998.

[HANDBOOK] Menezes, A., P. Van Oorschot and S. Vanstone, "Handbook of Applied Cryptography", CRC Press, 1997.

[MODES] Dworkin, M., "Recommendation for Block Cipher Modes of Operation: Methods and Techniques," NIST Special Publication 800-38A, December 2001. http://csrc.nist.gov/publications/nistpubs/800-38a/sp800-38a.pdf

[RFC-2026] Bradner, S., "The Internet Standards Process -- Revision 3", BCP 9, RFC 2026, October 1996.

[ROADMAP] Thayer, R., N. Doraswamy, and R. Glenn, "IP Security Document Roadmap", RFC 2411, November 1998.

[XCBC-MAC-2] Rogaway, Phil, email communications, October 2001.

10. Authors' Addresses

Sheila Frankel
NIST - National Institute of Standards and Technology
820 West Diamond Ave.
Room 677
Gaithersburg, MD 20899

Phone: +1 (301) 975-3297
EMail: sheila.frankel@nist.gov

Howard C. Herbert
Intel Corporation
Lan Access Division
5000 West Chandler Blvd.
MS-CH7-404
Chandler, Arizona 85226

Phone: +1 (480) 554-3116
EMail: howard.c.herbert@intel.com

11. Full Copyright Statement

Acknowledgement

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