Network Working Group	D. McGrew
Internet-Draft	Cisco Systems, Inc.
Intended status: Standards Track	January 12, 2011
Expires: July 16, 2011	

The use of AES-192 and AES-256 in Secure RTP
draft-ietf-avt-srtp-big-aes-06.txt

Abstract

This memo describes the use of the Advanced Encryption Standard (AES) with 192 and 256 bit keys within the Secure RTP protocol. It details Counter Mode encryption for SRTP and SRTCP and a new SRTP Key Derivation Function (KDF) for AES-192 and AES-256.

Status of this Memo

Copyright Notice

Table of Contents

---

1. Introduction

TOC

This memo describes the use of the Advanced Encryption Standard (AES) [FIPS197] (, "The Advanced Encryption Standard (AES)," .) with 192 and 256 bit keys within the Secure RTP protocol [RFC3711] (Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, "The Secure Real-time Transport Protocol (SRTP)," March 2004.). Below those block ciphers are referred to as AES-192 and AES-256, respectively, and the use of AES with a 128 bit key is referred to as AES-128. This document describes Counter Mode encryption for SRTP and SRTCP and appropriate SRTP Key Derivation Functions for AES-192 and AES-256. It also defines new cryptosuites that use these new functions.

While AES-128 is widely regarded as more than adequately secure, some users may be motivated to adopt AES-192 or AES-256 due to a perceived need to purse a highly conservative security strategy. For instance, the Suite B profile requires AES-256 for the protection of TOP SECRET information [suiteB] (, "Suite B Cryptography," .). (Note that while the AES-192 and AES-256 encryption methods defined in this document use Suite B algorithms, the cryptosuites in this document use the HMAC-SHA-1 algorithm, which is not included in Suite B.) See Section 6 for more discussion of security issues.

The crypto functions described in this document are an addition to, and not a replacement for, the crypto functions defined in [RFC3711]

(Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, “The Secure Real-time Transport Protocol (SRTP),” March 2004.).

1.1. Conventions Used In This Document

TOC

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] (Bradner, S., “Key words for use in RFCs to Indicate Requirement Levels,” March 1997.).

---

2. AES-192 and AES-256 Encryption

TOC

Section 4.1.1 of [RFC3711] (Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, “The Secure Real-time Transport Protocol (SRTP),” March 2004.) defines AES counter mode encryption, which it refers to as AES_CM. This definition applies to all of the AES key sizes. In this note, AES-192 counter mode and AES-256 counter mode and are denoted as AES_192_CM and AES_256_CM respectively. In both of these ciphers, the plaintext inputs to the block cipher are formed as in AES_CM, and the block cipher outputs are processed as in AES_CM. The only difference in the processing is that AES_192_CM uses AES-192, and AES_256_CM uses AES-256. Both AES_192_CM and AES_256_CM use a 112-bit salt as an input, as does AES_CM.
For the convenience of the reader, the structure of the counter blocks in SRTP counter mode encryption is illustrated in Figure 1 (AES Counter Mode.), using the terminology from Section 4.1.1 of [RFC3711] (Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, “The Secure Real-time Transport Protocol (SRTP),” March 2004.). In this diagram, the symbol (+) denotes the bitwise exclusive-or operation, and the AES encrypt operation uses AES-128, AES-192, or AES-256 for AES_CM, AES_192_CM, and AES_256_CM, respectively. The field labeled b_c contains a block counter, the value of which increments once for each invocation of the "AES Encrypt" function. The SSRC field is part of the RTP header [RFC3550] (Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, “RTP: A Transport Protocol for Real-Time Applications,” July 2003.).

```
          one octet
           <-->
            0  1  2  3  4  5  6  7  8  9  10 11 12 13 14 15
           +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
           |00|00|00|00|   SSRC    |   packet index  | b_c |---+
           +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+   |
                                                               |
           +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+   v
           |                  salt (k_s)             |00|00|->(+)
           +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+   |
                                                               |
                                                               v
                                                        +-------------+
                                  encryption key (k_e) -> | AES encrypt |
                                                        +-------------+
                                                               |
           +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+   |
           |                keystream block                |<--+
           +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
```

Figure 1: AES Counter Mode.

---

3. The AES_192_CM_PRF and AES_256_CM_PRF Key Derivation Functions

TOC

Section 4.3.3 of [RFC3711] (Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, “The Secure Real-time Transport Protocol (SRTP),” March 2004.) defines an AES counter mode key derivation function, which it refers to as AES_CM PRF (and sometimes as AES-CM PRF). (That specification uses the term PRF, or pseudo-random function, interchangeably with the phrase "key derivation function".) This key derivation function can be used with any AES key size. In this note, the AES-192 counter mode PRF and AES-256 counter mode PRF are denoted as AES_192_CM_PRF and AES_256_CM_PRF respectively. In both of these PRFs, the plaintext inputs to the block cipher are formed as in the AES_CM PRF, and the block cipher outputs are processed as in the AES_CM PRF. The only difference in the processing is that AES_192_CM_PRF uses AES-192, and AES_256_CM_PRF uses AES-256. Both AES_192_CM_PRF and AES_256_CM_PRF use a 112-bit salt as an input, as does the AES_CM PRF. For the convenience of the reader, the structure of the counter blocks in SRTP counter mode key derivation is illustrated in Figure 2 (The AES counter mode Key Derivation Function), using the terminology from Section 4.3.3 of [RFC3711] (Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, “The Secure Real-time Transport Protocol (SRTP),” March 2004.). In this diagram, the symbol (+) denotes the bitwise exclusive-or operation, and the "AES Encrypt" operation uses AES-128, AES-192, or AES-256 for the AES_CM PRF, AES_192_CM_PRF, and AES_256_CM_PRF, respectively. The field "LB" contains the 8-bit constant "label" which is provided as an input to the key derivation function (and which is distinct for each type of key generated by that function). The field labeled b_c contains a block counter, the value of which increments once for each invocation of the "AES Encrypt" function. The DIV operation is defined in Section 4.3.1 of [RFC3711] (Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, “The Secure Real-time Transport Protocol (SRTP),” March 2004.) as follows. Let "a DIV t" denote integer division of a by t, rounded down, and with the convention that "a DIV 0 = 0" for all a. We also make the convention of treating "a DIV t" as a bit string of the same length as a, and thus "a DIV t" will in general have leading zeros.

```
  one octet
   <-->
    0  1  2  3  4  5  6  7  8  9  10 11 12 13 14 15
   +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
   |00|00|00|00|00|00|00|LB| index DIV kdr   | b_c |---+
   +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+   |
                                                       |
   +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+   v
   |               master salt               |00|00|->(+)
   +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+   |
                                                       |
                                                       v
                                                +-------------+
                                  master key -> | AES encrypt |
                                                +-------------+
                                                       |
   +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+   |
   |                   output block                |<--+
   +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
```

Figure 2: The AES counter mode Key Derivation Function

TOC

3.1. Usage Requirements

When AES_192_CM is used for encryption, AES_192_CM_PRF SHOULD be used as the key derivation function, and AES_128_CM_PRF MUST NOT be used as the key derivation function.
When AES_256_CM is used for encryption, AES_256_CM_PRF SHOULD be used as the key derivation function. Both AES_128_CM_PRF and AES_192_CM_PRF MUST NOT be used as the key derivation function.
AES_256_CM_PRF MAY be used as the key derivation function when AES_CM is used for encryption, and when AES_192_CM is used for encryption.
AES_192_CM_PRF MAY be used as the key derivation function when AES_CM is used for encryption.

> Rationale: it is essential that the cryptographic strength of the key derivation meets or exceeds that of the encryption method. It is natural to use the same function for both encryption and key derivation. However, it is not required to do so because it is desirable to allow these ciphers to be used with alternative key derivation functions that may be defined in the future.

---

4. Crypto Suites

TOC

This section defines SRTP crypto suites that use the ciphers and key derivation functions defined in this document. The parameters in these crypto suites are described in Section 8.2 of [RFC3711] (Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, “The Secure Real-time Transport Protocol (SRTP),” March 2004.). These suites are registered with IANA for use with the SDP Security Descriptions attributes (Section 10.3.2.1 of [RFC4568] (Andreasen, F., Baugher, M., and D. Wing, “Session Description Protocol (SDP) Security Descriptions for Media Streams,” July 2006.)). Other SRTP key management methods that use the crypto functions defined in this document are encouraged to also use these crypto suite definitions.

> Rationale: the crypto suites use the same authentication function that is mandatory-to-implement in SRTP, HMAC-SHA1 with a 160 bit key. HMAC-SHA1 would accept larger key sizes, but when it is used with keys larger than 160 bits, it does not provide resistance to cryptanalysis greater than that security level, because it has only 160 bits of internal state. By retaining 160-bit authentication keys, the crypto suites in this note have more compatibility with existing crypto suites and implementations of them.

Parameter	Value
Master key length	192 bits
Master salt length	112 bits
Key Derivation Function	AES_192_CM_PRF (Section 3 (The AES_192_CM_PRF and AES_256_CM_PRF Key Derivation Functions))
Default key lifetime	2^31 packets
Cipher (for SRTP and SRTCP)	AES_192_CM (Section 2 (AES-192 and AES-256 Encryption))
SRTP authentication function	HMAC-SHA1 (Section 4.2.1 of [RFC3711] (Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, "The Secure Real-time Transport Protocol (SRTP)," March 2004.))
SRTP authentication key length	160 bits
SRTP authentication tag length	80 bits
SRTCP authentication function	HMAC-SHA1 (Section 4.2.1 of [RFC3711] (Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, "The Secure Real-time Transport Protocol (SRTP)," March 2004.))
SRTCP authentication key length	160 bits
SRTCP authentication tag length	80 bits

Table 1: The AES_192_CM_HMAC_SHA1_80 cryptosuite.

Parameter	Value
Master key length	192 bits
Master salt length	112 bits
Key Derivation Function	AES_192_CM_PRF (Section 3 (The AES_192_CM_PRF and AES_256_CM_PRF Key Derivation Functions))
Default key lifetime	2^31 packets

Cipher (for SRTP and SRTCP)	AES_192_CM (Section 2 (AES-192 and AES-256 Encryption))
SRTP authentication function	HMAC-SHA1 (Section 4.2.1 of [RFC3711] (Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, "The Secure Real-time Transport Protocol (SRTP)," March 2004.))
SRTP authentication key length	160 bits
SRTP authentication tag length	32 bits
SRTCP authentication function	HMAC-SHA1 (Section 4.2.1 of [RFC3711] (Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, "The Secure Real-time Transport Protocol (SRTP)," March 2004.))
SRTCP authentication key length	160 bits
SRTCP authentication tag length	80 bits

Table 2: The AES_192_CM_HMAC_SHA1_32 cryptosuite.

Parameter	Value
Master key length	256 bits
Master salt length	112 bits
Key Derivation Function	AES_256_CM_PRF (Section 3 (The AES_192_CM_PRF and AES_256_CM_PRF Key Derivation Functions))
Default key lifetime	2^31 packets
Cipher (for SRTP and SRTCP)	AES_256_CM (Section 2 (AES-192 and AES-256 Encryption))
SRTP authentication function	HMAC-SHA1 (Section 4.2.1 of [RFC3711] (Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, "The Secure Real-time Transport Protocol (SRTP)," March 2004.))
	160 bits

SRTP authentication key length	
SRTP authentication tag length	80 bits
SRTCP authentication function	HMAC-SHA1 (Section 4.2.1 of [RFC3711] (Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, “The Secure Real-time Transport Protocol (SRTP),” March 2004.))
SRTCP authentication key length	160 bits
SRTCP authentication tag length	80 bits

Table 3: The AES_256_CM_HMAC_SHA1_80 cryptosuite.

Parameter	**Value**
Master key length	256 bits
Master salt length	112 bits
Key Derivation Function	AES_256_CM_PRF (Section 3 (The AES_192_CM_PRF and AES_256_CM_PRF Key Derivation Functions))
Default key lifetime	2^31 packets
Cipher (for SRTP and SRTCP)	AES_256_CM (Section 2 (AES-192 and AES-256 Encryption))
SRTP authentication function	HMAC-SHA1 (Section 4.2.1 of [RFC3711] (Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, “The Secure Real-time Transport Protocol (SRTP),” March 2004.))
SRTP authentication key length	160 bits
SRTP authentication tag length	32 bits
	HMAC-SHA1 (Section 4.2.1 of [RFC3711] (Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K.

SRTCP authentication function	Norrman, "The Secure Real-time Transport Protocol (SRTP)," March 2004.))
SRTCP authentication key length	160 bits
SRTCP authentication tag length	80 bits

Table 4: The AES_256_CM_HMAC_SHA1_32 cryptosuite.

5. IANA Considerations

TOC

IANA is expected to assign the following parameters in the Session Description Protocol (SDP) Security Descriptions registry.

Crypto Suite Name	Reference
AES_192_CM_HMAC_SHA1_80	[RFCxxxx]
AES_192_CM_HMAC_SHA1_32	[RFCxxxx]
AES_256_CM_HMAC_SHA1_80	[RFCxxxx]
AES_256_CM_HMAC_SHA1_32	[RFCxxxx]

Note to RFC Editor: Replace RFCxxx with the number of this RFC.

6. Security Considerations

TOC

AES-128 provides a level of security that is widely regarded as being more than sufficient for providing confidentiality. It is believed that the economic cost of breaking AES-128 is significantly higher than the cost of more direct approaches to violating system security, e.g. theft, bribery, wiretapping, and other forms of malfeasance. Future advances in the state of the art of cryptanalysis could eliminate this confidence in AES-128, and motivate the use of AES-192 or AES-256. AES-192 is regarded as being secure even against some adversaries for which breaking AES-128 may be feasible. Similarly, AES-256 is regarded as being secure even against some adversaries for which it may be feasible to break AES-192. The availability of the larger key size versions of AES provides a fallback plan in case of unanticipated cryptanalytic results.

It is conjectured that AES-256 provides adequate security even against adversaries that possess the ability to construct a quantum computer that works on 256 or more quantum bits. No such computer is known to exist; its feasibility is an area of active speculation and research.

Despite the apparent sufficiency of AES-128, some users are interested in the larger AES key sizes. For some applications, the 40% increase in computational cost for AES-256 over AES-128 is a worthwhile bargain when traded for the security advantages outlined above. These applications include those with a perceived need for very high security, e.g. due to a desire for very long-term confidentiality.

AES-256 (as it is used in this note) provides the highest level of security, and it SHOULD be used whenever the highest possible security is desired. AES-192 provides a middle ground between the 128-bit and 256-bit versions of AES, and it MAY be used when security higher than that of AES-128 is desired. In this note, AES-192 and AES-256 are used with keys that are generated via a strong pseudorandom source, and thus the related-key attacks that have been described in the theoretical literature are not applicable.

As with any cipher, the conjectured security level of AES may change over time. The considerations in this section reflect the best knowledge available at the time of publication of this document.

It is desirable that AES_192_CM and AES_192_CM_PRF be used with an authentication function that uses a 192 bit key, and that AES_256_CM and AES_256_CM_PRF be used with an authentication function that uses a 256 bit key. However, this desire is not regarded as security-critical. Cryptographic authentication is resilient against future advances in cryptanalysis, since the opportunity for a forgery attack against a session closes when that session closes. For this reason, this note defines new ciphers, but not new authentication functions.

---

7. Test Cases

TOC

The test cases in this section are based on Appendix B of [RFC3711] (Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, “The Secure Real-time Transport Protocol (SRTP),” March 2004.).

7.1. AES-256-CM Test Cases

TOC

```
Keystream segment length: 1044512 octets (65282 AES blocks)
Session Key:      57f82fe3613fd170a85ec93c40b1f092
                  2ec4cb0dc025b58272147cc438944a98
Rollover Counter: 00000000
Sequence Number:  0000
SSRC:             00000000
Session Salt:     f0f1f2f3f4f5f6f7f8f9fafbfcfd0000 (already shifted)
Offset:           f0f1f2f3f4f5f6f7f8f9fafbfcfd0000

Counter                            Keystream

f0f1f2f3f4f5f6f7f8f9fafbfcfd0000   92bdd28a93c3f52511c677d08b5515a4
f0f1f2f3f4f5f6f7f8f9fafbfcfd0001   9da71b2378a854f67050756ded165bac
f0f1f2f3f4f5f6f7f8f9fafbfcfd0002   63c4868b7096d88421b563b8c94c9a31
...                                ...
f0f1f2f3f4f5f6f7f8f9fafbfcfdfeff   cea518c90fd91ced9cbb18c078a54711
f0f1f2f3f4f5f6f7f8f9fafbfcfdff00   3dbc4814f4da5f00a08772b63c6a046d
f0f1f2f3f4f5f6f7f8f9fafbfcfdff01   6eb246913062a16891433e97dd01a57f
```

7.2. AES_256_CM_PRF Test Cases

TOC

This section provides test data for the AES_256_CM_PRF key derivation function, which uses AES-256 in Counter Mode. In the following, we walk through the initial key derivation for the AES-256 Counter Mode cipher, which requires a 32 octet session encryption key and a 14 octet session salt, and the HMAC-SHA1 authentication function, which requires a 20-octet session authentication key. These values are called the cipher key, the cipher salt, and the auth key in the following. Since this is the initial key derivation and the key derivation rate is equal to zero, the value of (index DIV key_derivation_rate) is zero (actually, a six-octet string of zeros). In the following, we shorten key_derivation_rate to kdr.
The inputs to the key derivation function are the 32 octet master key and the 14 octet master salt:

```
master key:  f0f04914b513f2763a1b1fa130f10e29
             98f6f6e43e4309d1e622a0e332b9f1b6
master salt: 3b04803de51ee7c96423ab5b78d2
```

We first show how the cipher key is generated. The input block for AES-256-CM is generated by exclusive-oring the master salt with the concatenation of the encryption key label 0x00 with (index DIV kdr), then padding on the right with two null octets (which implements the multiply-by-2^16 operation, see Section 4.3.3 of RFC 3711). The resulting value is then AES-256-CM- encrypted using the master key to get the cipher key.

```
index DIV kdr:                 000000000000
label:                       00
master salt:   3b04803de51ee7c96423ab5b78d2
----------------------------------------------
xor:           3b04803de51ee7c96423ab5b78d2     (x, PRF input)

x*2^16:        3b04803de51ee7c96423ab5b78d20000 (AES-256-CM input)
x*2^16 + 1:    3b04803de51ee7c96423ab5b78d20001 (2nd AES input)

cipher key:    5ba1064e30ec51613cad926c5a28ef73 (1st AES output)
               1ec7fb397f70a960653caf06554cd8c4 (2nd AES output)
```

Next, we show how the cipher salt is generated. The input block for AES-256-CM is generated by exclusive-oring the master salt with the concatenation of the encryption salt label. That value is padded and encrypted as above.

```
index DIV kdr:                 000000000000
label:                       02
master salt:   3b04803de51ee7c96423ab5b78d2

----------------------------------------------
xor:           3b04803de51ee7cb6423ab5b78d2     (x, PRF input)

x*2^16:        3b04803de51ee7cb6423ab5b78d20000 (AES-256-CM input)

               fa31791685ca444a9e07c6c64e93ae6b (AES-256 ouptut)

cipher salt:   fa31791685ca444a9e07c6c64e93
```

We now show how the auth key is generated. The input block for AES-256-CM is generated as above, but using the authentication key label.

```
   index DIV kdr:                   000000000000
   label:                         01
   master salt:     3b04803de51ee7c96423ab5b78d2
   ----------------------------------------------
   xor:             3b04803de51ee7c86423ab5b78d2     (x, PRF input)

   x*2^16:          3b04803de51ee7c86423ab5b78d20000 (AES-256-CM in)
```

Below, the AES-256 output blocks that form the auth key are shown on the left, while the corresponding AES-256 input blocks are shown on the right. Note that the final AES-256 output is truncated to a four-byte length. The final auth key is shown below.

```
auth key blocks                    AES-256 input blocks
fd9c32d39ed5fbb5a9dc96b30818454d   3b04803de51ee7c86423ab5b78d20000
1313dc05                           3b04803de51ee7c86423ab5b78d20001

auth key: fd9c32d39ed5fbb5a9dc96b30818454d1313dc05
```

7.3. AES-192-CM Test Cases

TOC

```
 Keystream segment length: 1044512 octets (65282 AES blocks)
 Session Key:      eab234764e517b2d3d160d587d8c8621
                   9740f65f99b6bcf7
 Rollover Counter: 00000000
 Sequence Number:  0000
 SSRC:             00000000
 Session Salt:     f0f1f2f3f4f5f6f7f8f9fafbfcfd0000 (already shifted)
 Offset:           f0f1f2f3f4f5f6f7f8f9fafbfcfd0000

 Counter                            Keystream

 f0f1f2f3f4f5f6f7f8f9fafbfcfd0000   35096cba4610028dc1b57503804ce37c
 f0f1f2f3f4f5f6f7f8f9fafbfcfd0001   5de986291dcce161d5165ec4568f5c9a
 f0f1f2f3f4f5f6f7f8f9fafbfcfd0002   474a40c77894bc17180202272a4c264d
 ...                                ...
 f0f1f2f3f4f5f6f7f8f9fafbfcfdfeff   d108d1a31a00bad6367ec23eb044b415
 f0f1f2f3f4f5f6f7f8f9fafbfcfdff00   c8f57129fdeb970b59f917b257662d4c
 f0f1f2f3f4f5f6f7f8f9fafbfcfdff01   a5dab625811034e8cebdfeb6dc158dd3
```

7.4. AES_192_CM_PRF Test Cases

TOC

This section provides test data for the AES_192_CM_PRF key derivation function, which uses AES-192 in Counter Mode. In the following, we walk through the initial key derivation for the AES-192 Counter Mode cipher, which requires a 24 octet session encryption key and a 14 octet session salt, and the HMAC-SHA1 authentication function, which requires a 20-octet session authentication key. These values are called the cipher key, the cipher salt, and the auth key in the following. Since this is the initial key derivation and the key derivation rate is equal to zero, the value of (index DIV key_derivation_rate) is zero (actually, a six-octet string of zeros). In the following, we shorten key_derivation_rate to kdr.
The inputs to the key derivation function are the 24 octet master key and the 14 octet master salt:

```
master key:  73edc66c4fa15776fb57f9505c171365
             50ffda71f3e8e5f1
master salt: c8522f3acd4ce86d5add78edbb11
```

We first show how the cipher key is generated. The input block for AES-192-CM is generated by exclusive-oring the master salt with the concatenation of the encryption key label 0x00 with (index DIV kdr), then padding on the right with two null octets (which implements the multiply-by-2^16 operation, see Section 4.3.3 of RFC 3711). The resulting value is then AES-192-CM encrypted using the master key to get the cipher key.

```
index DIV kdr:                 000000000000
label:                       00
master salt:   c8522f3acd4ce86d5add78edbb11
-----------------------------------------------
xor:           c8522f3acd4ce86d5add78edbb11     (x, PRF input)

x*2^16:        c8522f3acd4ce86d5add78edbb110000 (AES-192-CM input)
x*2^16 + 1:    c8522f3acd4ce86d5add78edbb110001 (2nd AES input)

cipher key:    31874736a8f1143870c26e4857d8a5b2 (1st AES output)
               c4a354407faadabb                 (2nd AES output)
```

Next, we show how the cipher salt is generated. The input block for AES-192-CM is generated by exclusive-oring the master salt with the concatenation of the encryption salt label. That value is padded and encrypted as above.

```
index DIV kdr:                 000000000000
label:                       02
master salt:   c8522f3acd4ce86d5add78edbb11

----------------------------------------------
xor:           c8522f3acd4ce86f5add78edbb11     (x, PRF input)

x*2^16:        c8522f3acd4ce86f5add78edbb110000 (AES-192-CM input)

               2372b82d639b6d8503a47adc0a6c2590 (AES-192 ouptut)

cipher salt:   2372b82d639b6d8503a47adc0a6c
```

We now show how the auth key is generated. The input block for AES-192-CM is generated as above, but using the authentication key label.

```
index DIV kdr:                   000000000000
label:                         01
master salt:     c8522f3acd4ce86d5add78edbb11
-----------------------------------------------
xor:             c8522f3acd4ce86c5add78edbb11     (x, PRF input)

x*2^16:          c8522f3acd4ce86c5add78edbb110000 (AES-192-CM in)
```

Below, the AES-192 output blocks that form the auth key are shown on the left, while the corresponding AES-192 input blocks are shown on the right. Note that the final AES-192 output is truncated to a four-byte length. The final auth key is shown below.

```
auth key blocks                    AES-192 input blocks
355b10973cd95b9eacf4061c7e1a7151   c8522f3acd4ce86c5add78edbb110000
e7cfbfcb                           c8522f3acd4ce86c5add78edbb110001

auth key: 355b10973cd95b9eacf4061c7e1a7151e7cfbfcb
```

8. Acknowledgements

TOC

Thanks are due to John Mattsson for verifying the test cases in the document and providing comments, to Bob Bell for feedback and encouragement, and to Richard Barnes and Hilarie Orman for constructive review.

TOC

9. References

9.1. Normative References

TOC

[FIPS197]	"The Advanced Encryption Standard (AES)," FIPS-197 Federal Information Processing Standard.
[RFC2119]	Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels," BCP 14, RFC 2119, March 1997 (TXT, HTML, XML).
[RFC3550]	Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications," STD 64, RFC 3550, July 2003 (TXT, PS, PDF).
[RFC3711]	Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, "The Secure Real-time Transport Protocol (SRTP)," RFC 3711, March 2004 (TXT).
[RFC4568]	Andreasen, F., Baugher, M., and D. Wing, "Session Description Protocol (SDP) Security Descriptions for Media Streams," RFC 4568, July 2006 (TXT).

9.2. Informative References

TOC

[suiteB]	"Suite B Cryptography," http://www.nsa.gov/ia/programs/suiteb_cryptography/index.shtml.

Author's Address

TOC

	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