Network Working Group	A. Kato	
Internet-Draft	S. Kanno	
Intended status: Standards Track	NTT Software Corporation	
Expires: June 24, 2010	M. Kanda	
	NTT	
	T. Iwata	
	Nagoya University	
	December 21, 2009	

TOC

The Camellia-CMAC-96 and Camellia-CMAC-PRF-128 Algorithms and Its Use with IPsec
draft-kato-ipsec-camellia-cmac96and128-05

Abstract

This memo specifies two new algorithms for IPsec. One is the usage of Cipher-based Message Authentication Code (CMAC) with the Camellia block cipher as an authentication mechanism in the IPsec Encapsulating Security Payload and Authentication Header protocols. This algorithm is called Camellia-CMAC-96. The other algorithm is a pseudo-random function (PRF) based on CMAC with the Camellia block cipher for the Internet Key Exchange, version 2. This algorithm is called Camellia-CMAC-PRF-128.

Status of this Memo

Table of Contents

TOC

1. Introduction

The NIST specification [1] (National Institute of Standards and Technology, “Recommendation for Block Cipher Modes of Operation:The CMAC Mode for Authentication,” May 2005.) for the CMAC (Cipher-based Message Authentication Code) mode of operation for a block cipher describes a method to use the Advanced Encryption Standard (AES) as a Message Authentication Code (MAC) that has a 128-bit output length. The Camellia block cipher has the same external interface as the AES. This memo specifies two new algorithms for IPsec that replace AES by Camellia in already specified applications of CMAC for IPsec [8] (Song, JH., Poovendran, R., and J. Lee, “The AES-CMAC-96 Algorithm and Its Use with IPsec,” June 2006.) and [9] (Gustin, J. and A. Goyens, “A Uniform Resource Name (URN) Namespace for SWIFT Financial Messaging,” September 2003.). One is the usage of CMAC mode with Camellia [2] (Matsui, M., Nakajima, J., and S. Moriai, “A Description of the Camellia Encryption Algorithm,” April 2004.) as the underlying block cipher as an authentication mechanism in the IPsec Encapsulating Security Payload (ESP) [5] (Kent, S., “IP Encapsulating Security Payload (ESP),” December 2005.) and Authentication Header (AH) [4] (Kent, S., “IP Authentication Header,” December 2005.) protocols. This algorithm is called Camellia-CMAC-96. The 128-bit CMAC output is also useful as a long-lived Pseudo-Random Function (PRF). Thus, the other algorithm is a PRF based on CMAC with the Camellia block cipher for version 2 of the Internet Key Exchange (IKEv2) [6] (Kaufman, C., Hoffman, P., Nir, Y., and P. Eronen, “Internet Key Exchange Protocol: IKEv2,” April 2010.) that supports fixed and variable key sizes for the Key Derivation Function (KDF) and authentication, respectively. This algorithm is called Camellia-CMAC-PRF-128.
The Camellia algorithm and its properties are described in [2] (Matsui, M., Nakajima, J., and S. Moriai, “A Description of the Camellia Encryption Algorithm,” April 2004.). This document does not cover implementation details of CMAC. Those details can be found in [1] (National Institute of Standards and Technology, “Recommendation for Block Cipher Modes of Operation:The CMAC Mode for Authentication,” May 2005.). For further information on IKE, AH and ESP, refer to [3] (Kent, S. and K. Seo, “Security Architecture for the Internet Protocol,” December 2005.), [4] (Kent, S., “IP Authentication Header,” December 2005.), [5] (Kent, S., “IP Encapsulating Security Payload (ESP),” December 2005.), and [6] (Kaufman, C., Hoffman, P., Nir, Y., and P. Eronen, “Internet Key Exchange Protocol: IKEv2,” April 2010.).

---

2. Definitions

TOC

CBC Cipher Block Chaining mode of operation for a block cipher, providing a block cipher for arbitrary message sizes.

MAC

Message Authentication Code. A bit string of a fixed length, computed by the MAC generation algorithm, that is used to establish the authority and, hence, the integrity of a message.

CMAC A MAC algorithm based on a symmetric key block cipher in CBC mode, as specified in [1] (National Institute of Standards and Technology, "Recommendation for Block Cipher Modes of Operation:The CMAC Mode for Authentication," May 2005.).

Key (K) 128-bit (16-octet) key for the Camellia block cipher. Denoted by K.

Variable-length Key (VK) Variable-length key for Camellia-CMAC-PRF-128, denoted by VK.

Message (M) Message to be authenticated. Denoted by M.

Length (len) The length of message M in octets. Denoted by len. The minimum value is 0. The maximum value is not specified in this document.

VKlen The length of VK in octets.

truncate(T,l) Truncate T (MAC) in most-significant-bit-first (MSB-first) order to a length of l octets.

T The output of Camellia-CMAC.

Camellia-CMAC CMAC generation function based on the Camellia block cipher with 128-bit key.

Camellia-CMAC-96 IPsec AH and ESP MAC generation function based on Camellia-CMAC, which truncates the 128-bit CMAC output to its 96 most-significant bits.

Camellia-CMAC-PRF-128 IPsec AH and ESP PRF based on Camellia-CMAC, which removes the 128-bit key length restriction.

SKEYSEED Seed of shared key calculated from the nonces exchanged during the IKE_SA_INIT exchange and the Diffie-Hellman shared secret in the IKEv2 specification [6] (Kaufman, C., Hoffman, P., Nir, Y., and P. Eronen, "Internet Key Exchange Protocol: IKEv2," April 2010.).

TOC

3. Camellia-CMAC

The National Institute of Standards and Technology (NIST) has specified the Cipher-based Message Authentication Code (CMAC) [1] (National Institute of Standards and Technology, “Recommendation for Block Cipher Modes of Operation:The CMAC Mode for Authentication,” May 2005.). CMAC is a keyed hash function that is based on a symmetric key block cipher, such as the Advanced Encryption Standard [10] (National Institute of Standards and Technology, “Advanced Encryption Standard (AES),” November 2001.). The CMAC algorithm provides a framework for inserting various block cipher algorithms.

Camellia-CMAC uses the Camellia block cipher [2] (Matsui, M., Nakajima, J., and S. Moriai, “A Description of the Camellia Encryption Algorithm,” April 2004.) as a building block in CMAC [1] (National Institute of Standards and Technology, “Recommendation for Block Cipher Modes of Operation:The CMAC Mode for Authentication,” May 2005.). To generate a MAC, Camellia-CMAC(K, M, len) takes a secret key 'K', a message of variable length 'M', and the length of the message in octets 'len' as inputs and returns a fixed-length bit string.

In this specification, Camellia-CMAC is always used with 128-bit length key.

---

4. Camellia-CMAC-96

TOC

For IPsec message authentication in AH and ESP, Camellia-CMAC is used in its truncated form denoted as Camellia-CMAC-96 -- Camellia-CMAC with its output truncated to 96 bits. Its output is a 96-bit MAC that meets the default authenticator length as specified in [4] (Kent, S., “IP Authentication Header,” December 2005.). The result of truncation is taken in MSB-first order.

Figure 1 (Algorithm Camellia-CMAC-96) describes Camellia-CMAC-96 algorithm:

In step 1, Camellia-CMAC with key K is applied to the message M of length len octets.

In step 2, the output of step 1 is truncated to its first 12 octets, and the result (TT) is returned.

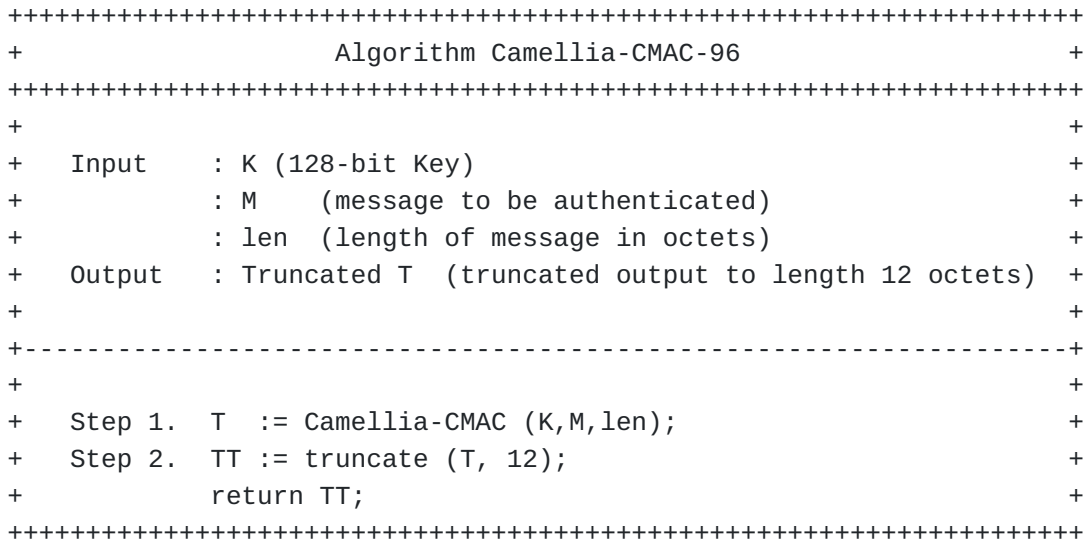

Figure 1: Algorithm Camellia-CMAC-96

5. Camellia-CMAC-PRF-128

TOC

The Camellia-CMAC-PRF-128 algorithm is identical to Camellia-CMAC except that the 128-bit key length restriction is removed.
IKEv2 [6] (Kaufman, C., Hoffman, P., Nir, Y., and P. Eronen, “Internet Key Exchange Protocol: IKEv2,” April 2010.) uses PRFs for multiple purposes, most notably for generating keying material and authentication of the IKE_SA.
When using Camellia-CMAC-PRF-128 as the PRF described in [6] (Kaufman, C., Hoffman, P., Nir, Y., and P. Eronen, “Internet Key Exchange Protocol: IKEv2,” April 2010.), Camellia-CMAC-PRF-128 is considered to admit a variable key length in all places, and the amount of keying material generated when new keys are generated is 128 bits (i.e., preferred key length when generating keying material of SK_d, SK_pi, and SK_pr is 128 bits).
When generating SKEYSEED the full of Ni and Nr are used as key for the PRF.

```
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+                          Camellia-CMAC-PRF-128                           +
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+                                                                          +
+ Input  : VK (Variable-length key)                                        +
+        : M (Message, i.e., the input data of the PRF)                    +
+        : VKlen (length of VK in octets)                                  +
+        : len (length of M in octets)                                     +
+ Output : PRV (128-bit Pseudo-Random Variable)                            +
+                                                                          +
+--------------------------------------------------------------------------+
+ Variable: K (128-bit key for Camellia-CMAC)                              +
+                                                                          +
+ Step 1.   If VKlen is equal to 16                                        +
+ Step 1a.  then                                                           +
+               K := VK;                                                   +
+ Step 1b.  else                                                           +
+               K := Camellia-CMAC(0^128, VK, VKlen);                      +
+ Step 2.   PRV := Camellia-CMAC(K, M, len);                               +
+           return PRV;                                                    +
+                                                                          +
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

Figure 2: Algorithm Camellia-CMAC-PRF-128

In step 1, the 128-bit key, K, for Camellia-CMAC is derived as follows:

- If the key VK is exactly 128 bits, then we use it as-is (step 1a).

- If it is longer or shorter than 128 bits, then we derive the key K by applying the Camellia-CMAC algorithm using the 128-bit all-zero string as the key and VK as the input message (step 1b).

In step 2, we apply the Camellia-CMAC algorithm using K as the key and M as the input message. The output of this algorithm is returned.

---

6. Test Cases

TOC

TOC

6.1. Camellia-CMAC-96

This section contains four test cases, which can be used to confirm that an implementation has correctly implemented Camellia-CMAC-96.

```
----------------
K               2b7e1516 28aed2a6 abf71588 09cf4f3c

Mlen = 0
M               <empty string>
T               ba925782 aaa1f5d9 a00f8964

----------------
K               2b7e1516 28aed2a6 abf71588 09cf4f3c

Mlen = 16
M               6bc1bee2 2e409f96 e93d7e11 7393172a
T               6d962854 a3b9fda5 6d7d45a9

----------------
K               2b7e1516 28aed2a6 abf71588 09cf4f3c

Mlen = 40
M               6bc1bee2 2e409f96 e93d7e11 7393172a
                ae2d8a57 1e03ac9c 9eb76fac 45af8e51
                30c81c46 a35ce411
T               5c18d119 ccd67661 44ac1866

----------------
K               2b7e1516 28aed2a6 abf71588 09cf4f3c

Mlen = 64
M               6bc1bee2 2e409f96 e93d7e11 7393172a
                ae2d8a57 1e03ac9c 9eb76fac 45af8e51
                30c81c46 a35ce411 e5fbc119 1a0a52ef
                f69f2445 df4f9b17 ad2b417b e66c3710
T               c2699a6e ba55ce9d 939a8a4e
```

6.2. Camellia-CMAC-PRF-128

TOC

This section contains twelve test cases, which can be used to confirm that an implementation has correctly implemented Camellia-CMAC-PRF-128. The first four test cases use 128 bit VK; the next four test cases use 192 bit VK; and the last four test cases use 256 bit VK.

```
VKlen = 16
 ----------------
 VK               2b7e1516 28aed2a6 abf71588 09cf4f3c


 Mlen = 0
 M                <empty string>
 T                ba925782 aaa1f5d9 a00f8964 8094fc71

 ----------------
 VK               2b7e1516 28aed2a6 abf71588 09cf4f3c


 Mlen = 16
 M                6bc1bee2 2e409f96 e93d7e11 7393172a
 T                6d962854 a3b9fda5 6d7d45a9 5ee17993

 ----------------
 VK               2b7e1516 28aed2a6 abf71588 09cf4f3c


 Mlen = 40
 M                6bc1bee2 2e409f96 e93d7e11 7393172a
                  ae2d8a57 1e03ac9c 9eb76fac 45af8e51
                  30c81c46 a35ce411
 T                5c18d119 ccd67661 44ac1866 131d9f22

 ----------------
 VK               2b7e1516 28aed2a6 abf71588 09cf4f3c


 Mlen = 64
 M                6bc1bee2 2e409f96 e93d7e11 7393172a
                  ae2d8a57 1e03ac9c 9eb76fac 45af8e51
                  30c81c46 a35ce411 e5fbc119 1a0a52ef
                  f69f2445 df4f9b17 ad2b417b e66c3710
 T                c2699a6e ba55ce9d 939a8a4e 19466ee9

 ------------------------------------------------------------
 VKlen = 24
 ----------------
 VK               8e73b0f7 da0e6452 c810f32b 809079e5
                  62f8ead2 522c6b7b

 K                abddaa68 e8b9f0af 2fb4db53 41cf1d91

 Mlen = 0
 M                <empty string>
 T                f4739892 c70bd23e 891f66c0 5fefbf27
```

```
----------------
VK               8e73b0f7 da0e6452 c810f32b 809079e5
                 62f8ead2 522c6b7b

K                abddaa68 e8b9f0af 2fb4db53 41cf1d91

Mlen = 16
M                6bc1bee2 2e409f96 e93d7e11 7393172a
T                60a33814 53babaed 1a11dfd3 d24c1410

----------------
VK               8e73b0f7 da0e6452 c810f32b 809079e5
                 62f8ead2 522c6b7b

K                abddaa68 e8b9f0af 2fb4db53 41cf1d91

Mlen = 40
M                6bc1bee2 2e409f96 e93d7e11 7393172a
                 ae2d8a57 1e03ac9c 9eb76fac 45af8e51
                 30c81c46 a35ce411
T                42b9d47f 4f58bc29 85b6f82c 23b121cb

----------------
VK               8e73b0f7 da0e6452 c810f32b 809079e5
                 62f8ead2 522c6b7b

K                abddaa68 e8b9f0af 2fb4db53 41cf1d91

Mlen = 64
M                6bc1bee2 2e409f96 e93d7e11 7393172a
                 ae2d8a57 1e03ac9c 9eb76fac 45af8e51
                 30c81c46 a35ce411 e5fbc119 1a0a52ef
                 f69f2445 df4f9b17 ad2b417b e66c3710
T                d078729f dcae9abc ff1ea4d6 18ed4501

------------------------------------------------------------
VKlen = 32
----------------
VK               603deb10 15ca71be 2b73aef0 857d7781
                 1f352c07 3b6108d7 2d9810a3 0914dff4

K                b5aeeae9 2c23bed7 167af194 2e831597

Mlen = 0
M                <empty string>
T                c96d7d40 d4aaab78 ac906b91 c82bd690

----------------
VK               603deb10 15ca71be 2b73aef0 857d7781
```

```
                  1f352c07 3b6108d7 2d9810a3 0914dff4

K                 b5aeeae9 2c23bed7 167af194 2e831597

Mlen = 16
M                 6bc1bee2 2e409f96 e93d7e11 7393172a
T                 104de4b9 0da6baf1 fa73945b e614f032

----------------
VK                603deb10 15ca71be 2b73aef0 857d7781
                  1f352c07 3b6108d7 2d9810a3 0914dff4

K                 b5aeeae9 2c23bed7 167af194 2e831597

Mlen = 40
M                 6bc1bee2 2e409f96 e93d7e11 7393172a
                  ae2d8a57 1e03ac9c 9eb76fac 45af8e51
                  30c81c46 a35ce411
T                 2d3684e9 1cb1b303 a7db8648 f25ee16c

----------------
VK                603deb10 15ca71be 2b73aef0 857d7781
                  1f352c07 3b6108d7 2d9810a3 0914dff4

K                 b5aeeae9 2c23bed7 167af194 2e831597

Mlen = 64
M                 6bc1bee2 2e409f96 e93d7e11 7393172a
                  ae2d8a57 1e03ac9c 9eb76fac 45af8e51
                  30c81c46 a35ce411 e5fbc119 1a0a52ef
                  f69f2445 df4f9b17 ad2b417b e66c3710
T                 d6b0f1b7 dda2b62a eca6d51d da63fdda
```

---

7. Security Considerations

TOC

The security provided by Camellia-CMAC-96 and Camellia-CMAC-PRF-128 is built on the strong cryptographic algorithm Camellia and CMAC. At the time of this writing, there are no known practical cryptographic attacks against Camellia or CMAC.
However, as is true with any cryptographic algorithm, part of its strength lies in the secret key, K, and the correctness of the implementation in all of the participating systems. If the secret key is compromised or inappropriately shared, it guarantees neither authentication nor message integrity at all. The secret key shall be generated in a way that meets the pseudo randomness requirement of RFC

4086 [7] (Eastlake, D., Schiller, J., and S. Crocker, “Randomness Requirements for Security,” June 2005.) and should be kept safe. If and only if Camellia-CMAC-96 and Camellia-CMAC-PRF-128 are used properly it provides the authentication and integrity that meet the best current practice of message authentication.

8. IANA Considerations

TOC

The IANA has allocated value <TBD1> for IKEv2 Transform Type 3 (Integrity Algorithm) to the AUTH_CAMELLIA_CMAC_96 algorithm, and has allocated a value of <TBD2> for IKEv2 Transform Type 2 (Pseudo-Random Function) to the PRF_CAMELLIA128_CMAC algorithm.

9. Acknowledgements

TOC

We thank Tim Polk and Tero Kivinen for their initial review of this document. Special thanks to Alfred Hoenes for several very detailed reviews and suggestions.

10. References

TOC

10.1. Normative

TOC

[1]	National Institute of Standards and Technology, “Recommendation for Block Cipher Modes of Operation:The CMAC Mode for Authentication,” Special Publication 800-38B, May 2005.
[2]	Matsui, M., Nakajima, J., and S. Moriai, “A Description of the Camellia Encryption Algorithm,” RFC 3713, April 2004 (TXT).

10.2. Informative

TOC

[3]	Kent, S. and K. Seo, “Security Architecture for the Internet Protocol,” RFC 4301, December 2005 (TXT).
[4]	Kent, S., “IP Authentication Header,” RFC 4302, December 2005 (TXT).
[5]	Kent, S., “IP Encapsulating Security Payload (ESP),” RFC 4303, December 2005 (TXT).

[6]	Kaufman, C., Hoffman, P., Nir, Y., and P. Eronen, “Internet Key Exchange Protocol: IKEv2,” draft-ietf-ipsecme-ikev2bis-10 (work in progress), April 2010 (TXT).
[7]	Eastlake, D., Schiller, J., and S. Crocker, “Randomness Requirements for Security,” BCP 106, RFC 4086, June 2005 (TXT).
[8]	Song, JH., Poovendran, R., and J. Lee, “The AES-CMAC-96 Algorithm and Its Use with IPsec,” RFC 4494, June 2006 (TXT).
[9]	Gustin, J. and A. Goyens, “A Uniform Resource Name (URN) Namespace for SWIFT Financial Messaging,” RFC 3615, September 2003 (TXT).
[10]	National Institute of Standards and Technology, “Advanced Encryption Standard (AES),” FIPS PUB 197, November 2001.

Authors' Addresses

TOC

	Akihiro Kato
	NTT Software Corporation
Phone:	+81-45-212-9803
Fax:	+81-45-212-9800
Email:	kato.akihiro@po.ntts.co.jp
	Satoru Kanno
	NTT Software Corporation
Phone:	+81-45-212-9803
Fax:	+81-45-212-9800
Email:	kanno.satoru@po.ntts.co.jp
	Masayuki Kanda
	NTT
Phone:	+81-422-59-3456
Fax:	+81-422-59-4015
Email:	kanda.masayuki@lab.ntt.co.jp
	Tetsu Iwata
	Nagoya University
Email:	iwata@cse.nagoya-u.ac.jp