CMAC-based Extract-and-Expand Key Derivation Function (CKDF)
draft-agl-ckdf-00

Abstract

This memo describes a KDF based on AES-CMAC.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

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."

This Internet-Draft will expire on February 22, 2016.

Copyright Notice

Copyright (c) 2015 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 (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.
1. Introduction

The HKDF key derivation function, described in [RFC5869], is currently the de-facto KDF for use in a variety of protocols. However, in hardware orientated designs, significant space savings can be achieved if the underlying primitive is AES rather than a cryptographic hash function.

The memo specifies CKDF, the CMAC-based Key Derivation Function. It is, succinctly, HKDF but with HMAC [RFC2104] replaced by CMAC [RFC4493].

2. CKDF

CKDF follows exactly the same structure as [RFC5869] but "HMAC-Hash" is replaced by the function "AES-CMAC" throughout. The "AES-CMAC" function also takes two arguments: the first is a 16 byte key and the second is an input. It returns the AES-CMAC MAC of the input using the given key as an AES key.

Thus, following HKDF, the "CKDF-Extract(salt, IKM)" function takes an optional, 16-byte salt and an arbitrary-length "input keying material" (IKM) message. If no salt is given, the 16-byte, all-zero value is used. It returns the result of "AES-CMAC(key = salt, input = IKM)", called the "pseudorandom key" (PRK), which will be 16 bytes long.

Likewise, the "CKDF-Expand(PRK, info, L)" function takes the PRK result from "CKDF-Extract", an arbitrary "info" argument and a requested number of bytes to produce. It calculates the L-byte result, called the "output keying material" (OKM), as:
\[ N = \text{ceil}(L/16) \]
\[ T = T(1) \, \mid \, T(2) \, \mid \, T(3) \, \mid \, \ldots \, \mid \, T(N) \]
\[ \text{OKM} = \text{first L octets of } T \]

where:
\[ T(0) = \text{empty string (zero length)} \]
\[ T(1) = \text{AES-CMAC(PRK, T(0) \, \mid \, info \, \mid \, 0x01)} \]
\[ T(2) = \text{AES-CMAC(PRK, T(1) \, \mid \, info \, \mid \, 0x02)} \]
\[ T(3) = \text{AES-CMAC(PRK, T(2) \, \mid \, info \, \mid \, 0x03)} \]
\[ \ldots \]

(where the constant concatenated to the end of each \( T(n) \) is a single octet.)

Note that AES-CMAC in [RFC4493] is only defined for AES-128 and likewise, so is CKDF. However, the dependency on AES-128 is stronger here because the length of the PRK from "CKDF-Extract" is the AES blocksize of 128 bits. Thus, if one wished to use AES-256 in the future, the PRK would, somehow, need to be 256 bits. Given the complexities of this, those wishing a higher security level should instead use HKDF with a suitable hash function.

3. Test Vectors

3.1. CKDF-Extract

This section contains test vectors for the "CKDF-Extract" function.

\# These two test vectors are from RFC4493, section 4
Salt: 2b7e1516 28aed2a6 abf71588 09cf4f3c  
IKM:  (empty)  
PRK:  bb1d6929 e9593728 7fa37d12 9b756746

Salt: 2b7e1516 28aed2a6 abf71588 09cf4f3c  
IKM:  6bc1bee2 2e409f96 e93d7e11 7393172a  
PRK:  070a16b4 6b4d4144 f79bdd9d d04a287c

Salt: (none)  
IKM:  73656372 6574206b 6579  
PRK:  6f79b401 ea761a01 00b7ca60 c178b69d

3.2. CKDF-Expand

This section contains test vectors for the "CKDF-Expand" function.
4. Security Considerations

Since CKDF is so closely based on HKDF, the security considerations are the same and sections 3, 4 and 5 of [RFC5869] are included here by reference.

5. IANA Considerations

None.

6. References

6.1. Normative References


6.2. Informative References

Author’s Address

Adam Langley
Google Inc

Email: agl@google.com