

TCPM Working Group
Internet-Draft
Intended status: Experimental
Expires: 29 November 2026

R. Bonica
T. Li
HPE
28 May 2026

384-bit Cryptographic Algorithms For Use With TCP-AO
draft-bonica-tcpm-tcp-ao-long-algs-01

Abstract

RFC5926 creates a list of cryptographic algorithms that can be used with TCP-AO. This document expands that list, adding two Key Derivation Functions (KDFs) and two Message Authentication Code (MAC) Algorithms. They are called HKDF-SHA384, KMAC384-KDF, HMAC-SHA384, and KMAC384.

The algorithms described by this document produce 384-bit (i.e., 48-byte) MACs. When 48-byte MACs are encoded in TCP-AO, the TCP-AO consumes 52 bytes. This exceeds TCP's current 40-byte option size limitation.

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 <https://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 29 November 2026.

Copyright Notice

Copyright (c) 2026 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 (<https://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 Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

Table of Contents

1. Introduction	2
2. Requirements Language	2
3. Algorithm Classes	2
3.1. Key Derivation Functions (KDFs)	3
3.1.1. HKDF-SHA384	3
3.1.2. KMAC384-KDF	4
3.2. MAC Algorithms	5
3.2.1. The Use of HMAC-SHA384	6
3.2.2. The Use of KMAC384	6
4. Security Considerations	7
5. IANA Considerations	7
6. Acknowledgements	7
7. Normative References	7
Authors' Addresses	8

1. Introduction

[RFC5926] creates a list of cryptographic algorithms that can be used with TCP-AO [RFC5925]. This document expands that list, adding two Key Derivation Functions (KDFs) and two Message Authentication Code (MAC) Algorithms. They are called HKDF-SHA384, KMAC384-KDF, HMAC-SHA384, and KMAC384.

The algorithms described by this document produce 384-bit (i.e., 48-byte) MACs. When 48-byte MACs are encoded in TCP-AO, the TCP-AO consumes 52 bytes. This exceeds TCP's current [RFC9293] 40-byte option size limitation.

2. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

3. Algorithm Classes

[RFC5925] requires the following cryptographic algorithm classes:

- * Key Derivation Functions (KDFs)

- * MAC Algorithms

Section 3.1 of this document addresses KDFs while Section 3.2 addresses MAC algorithms.

3.1. Key Derivation Functions (KDFs)

A KDF converts Initial Keying Material (IKM) into cryptographically secure Output Keying Material (OKM). In the case of TCP-AO, a KDF converts an administratively assigned Master_Key into a Traffic_Key.

KDFs have the following interface:

- * Traffic_Key = KDF_alg(Master_Key, Context, Output_Length)

where:

- * KDF_alg is the KDF algorithm being used.
- * Master_Key is a variable length, human-readable pre-shared key (PSK).
- * Context is binary string containing information related to the TCP connection, as defined in [RFC5925], Section 5.2.
- * Output_Length is the desired length of the Traffic_Key. In this document, the Output_Length is always equal to 384 bits.

This document defines two KDFs:

- * HKDF-SHA384
- * KMAC384-KDF

Section 3.1.1 of this document describes HKDF-SHA384 while Section 3.1.2 describes KMAC384-KDF.

3.1.1. HKDF-SHA384

HKDF-SHA384 is as described in [RFC5869]. HKDF-SHA384 executes in the following stages:

- * Extract
- * Expand

The interface to the Extract stage is:

* $PRK = HKDF\text{-}Extract(salt, IKM)$

where:

* PRK is a Pseudo-random key, to be used in the Expand stage.

* $salt$ is a 48-byte string of all-zero values.

* IKM is the `Master_Key` argument provided to the KDF interface.

According to [RFC5869], the goal of the extract stage is to concentrate the possibly dispersed entropy of the input keying material into a short, but cryptographically strong pseudorandom key. In some applications, the input may already be a good pseudorandom key. In these cases, the extract stage is not necessary, and the expand part can be used alone. However, when used with TCP-AO, implementations MUST execute the extract stage.

The interface to the Expand Step is:

* $OKM = HKDF\text{-}Expand(PRK, info, L)$

where:

* OKM is the `Traffic_Key`.

* PRK is the value produced by the Extract stage.

* $info$ is the `Context` argument provided to the KDF interface.

* L is the `Output_Length` argument provided to the KDF interface.

The expand step expands the pseudorandom key to the desired length. The number and lengths of the output keys depend on the specific cryptographic algorithms for which the keys are needed.

3.1.2. KMAC384-KDF

KMAC384-KDF is as described in [RFC9688]. The interface to KMAC384-KDF is:

* $OKM = KMAC384(K, X, L, S)$.

Where:

* OKM is the `Traffic_Key`.

* K is the `Master_Key` argument provided to the KDF interface.

- * X is the Context argument provided to the KDF interface.
- * L is the Output_Length argument provided to the KDF interface.
- * S is application specific information. It is always equal to "TCP-AO".

KMAC384-KDF operates in the following stages:

- * Absorb
- * Squeeze

Both stages compensate for lack of entropy in the Master_Key and neither stage is optional.

3.2. MAC Algorithms

Each MAC algorithm defined for TCP-AO has the following fixed elements as part of its definition:

- * KDF_Alg is the name of the KDF algorithm used to generate the Traffic_Key.
- * Key_Length is the length of the Traffic_Key used in this MAC, measured in bits. In this document, the Key_Length is always 384 bits.
- * MAC_Length is the desired length of the MAC to be produced by the algorithm. In this document, the MAC_Length is always 384 bits.

MACs computed for TCP-AO have the following interface:

- * $MAC = MAC_alg(Traffic_Key, Message)$

where:

- * MAC is the value to be encoded in TCP-AO.
- * MAC_alg is MAC Algorithm used.
- * Traffic_Key is the result of KDF.
- * Message is the message to be authenticated, as specified in [RFC5925], Section 5.1.

3.2.1. The Use of HMAC-SHA384

The three fixed elements for HMAC-SHA384 are:

- * KDF_Alg: HKDF-SHA384.
- * Key_Length: 384 bits.
- * MAC_Length: 384 bits.

For:

- * MAC = MAC_alg (Traffic_Key, Message)

HMAC-SHA-384 for TCP-AO has the following values:

- * MAC is the value to be encoded in TCP-AO.
- * MAC_alg is HMAC-SHA384.
- * Traffic_Key is the result of the KDF.
- * Message is the message to be authenticated, as specified in [RFC5925], Section 5.1.

3.2.2. The Use of KMAC384

The three fixed elements for KMAC384 are:

- * KDF_Alg: KMAC384-KDF.
- * Key_Length: 384 bits.
- * MAC_Length: 384 bits.

For:

- * MAC = MAC_alg (Traffic_Key, Message)

KMAC384 for TCP-AO has the following values:

- * MAC is the value to be encoded in TCP-AO.
- * MAC_alg is KMAC384.
- * Traffic_Key is the result of the KDF.

- * Message is the message to be authenticated, as specified in [RFC5925], Section 5.1.

4. Security Considerations

This document inherits all of the security considerations of [RFC5869], [RFC5925], and [RFC8702].

The security of cryptography-based systems depends on both the strength of the cryptographic algorithms chosen and the strength of the keys used with those algorithms. The security also depends on the engineering of the protocol used by the system to ensure that there are no non-cryptographic ways to bypass the security of the overall system.

Master_Keys SHOULD have 256 bits of entropy. This document RECOMMENDS that operators randomly generate 256-bit strings for use as Master_Keys. However, it is understood that may not do so.

TCP-AO Master Key Tuples MUST be rotated at a rate commensurate with the strength of the cryptographic algorithms.

5. IANA Considerations

IANA is requested to add the following entries to the "Cryptographic Algorithms for TCP-AO Registration" (<https://www.iana.org/assignments/tcp-parameters/tcp-parameters.xhtml#tcp-parameters-3>).

+=====+=====+		
Algorithm	Reference	
+=====+=====+		
HMAC-SHA384	This Document	
+-----+-----+		
KMAC384	This Document	
+-----+-----+		

Table 1: IANA Actions

6. Acknowledgements

Thanks to Eric Biggers, Lars Eggert, Gorrry Fairhurst, C.M. Heard, Russ Housley, John Mattsson, Yoshifumi Nishida, Joe Touch, Michael Tuxen, and Magnus Westerlund for their review and comments.

7. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/rfc/rfc2119>>.
- [RFC5869] Krawczyk, H. and P. Eronen, "HMAC-based Extract-and-Expand Key Derivation Function (HKDF)", RFC 5869, DOI 10.17487/RFC5869, May 2010, <<https://www.rfc-editor.org/rfc/rfc5869>>.
- [RFC5925] Touch, J., Mankin, A., and R. Bonica, "The TCP Authentication Option", RFC 5925, DOI 10.17487/RFC5925, June 2010, <<https://www.rfc-editor.org/rfc/rfc5925>>.
- [RFC5926] Lebovitz, G. and E. Rescorla, "Cryptographic Algorithms for the TCP Authentication Option (TCP-AO)", RFC 5926, DOI 10.17487/RFC5926, June 2010, <<https://www.rfc-editor.org/rfc/rfc5926>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/rfc/rfc8174>>.
- [RFC8702] Kampanakis, P. and Q. Dang, "Use of the SHAKE One-Way Hash Functions in the Cryptographic Message Syntax (CMS)", RFC 8702, DOI 10.17487/RFC8702, January 2020, <<https://www.rfc-editor.org/rfc/rfc8702>>.
- [RFC9293] Eddy, W., Ed., "Transmission Control Protocol (TCP)", STD 7, RFC 9293, DOI 10.17487/RFC9293, August 2022, <<https://www.rfc-editor.org/rfc/rfc9293>>.
- [RFC9688] Housley, R., "Use of the SHA3 One-Way Hash Functions in the Cryptographic Message Syntax (CMS)", RFC 9688, DOI 10.17487/RFC9688, November 2024, <<https://www.rfc-editor.org/rfc/rfc9688>>.

Authors' Addresses

Ron Bonica
HPE
United States of America
Email: ronald.bonica@hpe.com

Tony Li
HPE
United States of America

Email: tony.li@tony.li