TCPM Working Group
Internet-Draft
Intended status: Experimental
Expires: 24 September 2026

R. Bonica
T. Li
HPE

23 March 2026

Additional Security Algorithms For Use With TCP-AO
draft-bonica-tcpm-tcp-ao-long-algs-00

Abstract

RFC5926 specifies cryptographic algorithms for TCP-AO. It explains how to use KDF_HMAC_SHA1 and KDF_AES_128_CMAC as KDFs. It also explains how to use HMAC-SHA-1-96 and AES-128-CMAC-96 as MAC algorithms.

This document specifies several new KDFs and MAC algorithms for TCP-AO. The KDFs and MAC algorithms specified in this document use stronger cryptography.

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 24 September 2026.

Copyright Notice

Table of Contents

1. Introduction

TCP end-points use the TCP Authentication Option (TCP-AO) [RFC5925] to authenticate segments. TCP-AO relies upon:

* A Master Key Tuple (MKT)

* A Key Derivation Function (KDF)

* A Message Authentication Code (MAC) algorithm

TCP-AO systems are configured with one or more MKTs for each connection that they protect. When a connection is associated with multiple MKTs, TCP-AO can rotate among them during the course of a TCP session. This facilitates dynamic key change and authentication algorithm agility.

An MKT includes:

* Two MKT identifiers, one used for sending and one used for receiving

* A connection identifier (i.e., a TCP socket pair)

* A master key (i.e., a shared secret)

* A KDF

* A MAC algorithm

* A flag indicating whether TCP options other than TCP-AO are authenticated

The KDF generates a traffic key. Its inputs are:

* A pseudorandom function (PRF) used to generate the traffic key

* The master key

* Context (i.e., A binary string containing information related to the connection)

* Output length (i.e., the length of the traffic key, in bits)

The MAC algorithm produces a MAC. It is defined by:

* The KDF algorithm used to generate the traffic key

* The length of the traffic key, in bits

* The length of the MAC, in bits

The following are inputs to the MAC Algorithm:

* traffic key

* message

TCP-AO systems include the MAC in the TCP-AO. They use the MAC to authenticate segments.

[RFC5926] specifies cryptographic algorithms for TCP-AO. It explains how to use KDF_HMAC_SHA1 and KDF_AES_128_CMAC as KDFs. It also explains how to use HMAC-SHA-1-96 and AES-128-CMAC-96 as MAC algorithms.

This document specifies several new KDFs and MAC algorithms for TCP-AO. The KDFs and MAC algorithms defined in this document use stronger cryptography.

The MAC algorithms described this document yield MACs ranging from 256 to 512 bits (i.e., 32 to 64 bytes). Therefore, when they are encoded in a TCP-AO, the TCP-AO ranges from 36 to 68 bytes.

The TCP-AO is encoded in the TCP Options field. The TCP Options field is frequently required to carry multiple options, including the TCP-AO.

Currently, the TCP-Options field cannot exceed 40 bytes. However, TCP Extended Options [I-D.bonica-tcpm-extended-options] removes this limitation. Therefore, the MAC algorithms described in this document can only be used on systems that support TCP Extended Options or some other mechanism that extends the TCP Options field.

2. Conventions and Definitions

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. Updates to RFC 5926

3.1. Concrete KDFs

3.1.1. KDF_HMAC_SHA256

For KDF_HMAC_SHA256:

* PRF for KDF_alg: HMAC-SHA256 [RFC2104] [DOI.10.6028_NIST.FIPS.180-4]

* Use: HMAC-SHA256(Key, Input).

* Input: (i || Label || Context || Output_Length)

* Key: Master_Key, configured by user, and passed to the KDF

* Output_Length: 256 bits

* Result: Traffic_Key, used in the MAC function by TCP-AO

3.1.2. KDF_HMAC_SHA384

For KDF_HMAC_SHA384:

* PRF for KDF_alg: HMAC-SHA384 [RFC2104] [DOI.10.6028_NIST.FIPS.180-4]

* Use: HMAC-SHA384(Key, Input).

* Input: (i || Label || Context || Output_Length)

* Key: Master_Key, configured by user, and passed to the KDF

* Output_Length: 384 bits

* Result: Traffic_Key, used in the MAC function by TCP-AO

3.1.3. KDF_HMAC_SHA512

For KDF_HMAC_SHA512:

* PRF for KDF_alg: HMAC-SHA512 [RFC2104] [DOI.10.6028_NIST.FIPS.180-4]

* Use: HMAC-SHA512(Key, Input).

* Input: (i || Label || Context || Output_Length)

* Key: Master_Key, configured by user, and passed to the KDF

* Output_Length: 224 bits

* Result: Traffic_Key, used in the MAC function by TCP-AO

3.1.4. KDF_HMAC_SHA3-256

For KDF_HMAC_SHA3-256:

* PRF for KDF_alg: HMAC-SHA3-256 [RFC2104] [DOI.10.6028_NIST.FIPS.202]

* Use: HMAC-SHA3-256(Key, Input).

* Input: (i || Label || Context || Output_Length)

* Key: Master_Key, configured by user, and passed to the KDF

* Output_Length: 256 bits

* Result: Traffic_Key, used in the MAC function by TCP-AO

3.1.5. KDF_HMAC_SHA3-384

For KDF_HMAC_SHA3-384:

* PRF for KDF_alg: HMAC-SHA3-384 [RFC2104] [DOI.10.6028_NIST.FIPS.202]

* Use: HMAC-SHA3-384(Key, Input).

* Input: (i || Label || Context || Output_Length)

* Key: Master_Key, configured by user, and passed to the KDF

* Output_Length: 384 bits

* Result: Traffic_Key, used in the MAC function by TCP-AO

3.1.6. KDF_HMAC_SHA3-512

For KDF_HMAC_SHA3-512:

* PRF for KDF_alg: HMAC-SHA3-512 [RFC2104] [DOI.10.6028_NIST.FIPS.202]

* Use: HMAC-SHA3-512(Key, Input).

* Input: (i || Label || Context || Output_Length)

* Key: Master_Key, configured by user, and passed to the KDF

* Output_Length: 512 bits

* Result: Traffic_Key, used in the MAC function by TCP-AO

3.2. MAC Algorithms

3.2.1. The Use of HMAC-SHA256

By definition, HMAC [RFC2104] requires a cryptographic hash function. SHA256 will be that hash function used for authenticating and providing integrity validation on TCP segments with HMAC.

The three fixed elements for HMAC-SHA256 are:

* KDF_Alg: KDF_HMAC_SHA256

* Key_Length: 256 bits.

* MAC_Length: 256 bits.

For:

* MAC = MAC_alg (Traffic_Key, Message)

HMAC-SHA256 for TCP-AO has the following values:

* MAC_alg: HMAC-SHA256

* Traffic_Key: Variable; the result of the KDF.

* Message: The message to be authenticated, as specified in [RFC5925], Section 5.1.

3.2.2. The Use of HMAC-SHA384

By definition, HMAC [RFC2104] requires a cryptographic hash function. SHA384 will be that hash function used for authenticating and providing integrity validation on TCP segments with HMAC.

The three fixed elements for HMAC-SHA384 are:

* KDF_Alg: KDF_HMAC_SHA384

* Key_Length: 384 bits.

* MAC_Length: 384 bits.

For:

* MAC = MAC_alg (Traffic_Key, Message)

HMAC-SHA384 for TCP-AO has the following values:

* MAC_alg: HMAC-SHA384

* Traffic_Key: Variable; the result of the KDF.

* Message: The message to be authenticated, as specified in [RFC5925], Section 5.1.

3.2.3. The Use of HMAC-SHA512

By definition, HMAC [RFC2104] requires a cryptographic hash function. SHA512 will be that hash function used for authenticating and providing integrity validation on TCP segments with HMAC.

The three fixed elements for HMAC-SHA512 are:

* KDF_Alg: KDF_HMAC_SHA512

* Key_Length: 512 bits.

* MAC_Length: 512 bits.

For:

* MAC = MAC_alg (Traffic_Key, Message)

HMAC-SHA512 for TCP-AO has the following values:

* MAC_alg: HMAC-SHA512

* Traffic_Key: Variable; the result of the KDF.

* Message: The message to be authenticated, as specified in [RFC5925], Section 5.1.

3.2.4. The Use of HMAC-SHA3-256

By definition, HMAC [RFC2104] requires a cryptographic hash function. SHA3-256 will be that hash function used for authenticating and providing integrity validation on TCP segments with HMAC.

The three fixed elements for HMAC-SHA3-256 are:

* KDF_Alg: KDF_HMAC_SHA3-256.

* Key_Length: 256 bits.

* MAC_Length: 256 bits.

For:

* MAC = MAC_alg (Traffic_Key, Message)

HMAC-SHA3-256 for TCP-AO has the following values:

* MAC_alg: HMAC-SHA3-256.

* Traffic_Key: Variable; the result of the KDF.

* Message: The message to be authenticated, as specified in [RFC5925], Section 5.1.

3.2.5. The Use of HMAC-SHA3-384

By definition, HMAC [RFC2104] requires a cryptographic hash function. SHA3-384 will be that hash function used for authenticating and providing integrity validation on TCP segments with HMAC.

The three fixed elements for HMAC-SHA3-384 are:

* KDF_Alg: KDF_HMAC_SHA3-384.

* Key_Length: 384 bits.

* MAC_Length: 384 bits.

For:

* MAC = MAC_alg (Traffic_Key, Message)

HMAC-SHA3-384 for TCP-AO has the following values:

* MAC_alg: HMAC-SHA3-384.

* Traffic_Key: Variable; the result of the KDF.

* Message: The message to be authenticated, as specified in [RFC5925], Section 5.1.

3.2.6. The Use of HMAC-SHA3-512

By definition, HMAC [RFC2104] requires a cryptographic hash function. SHA3-512 will be that hash function used for authenticating and providing integrity validation on TCP segments with HMAC.

The three fixed elements for HMAC-SHA3-224 are:

* KDF_Alg: KDF_HMAC_SHA3-512.

* Key_Length: 512 bits.

* MAC_Length: 512 bits.

For:

* MAC = MAC_alg (Traffic_Key, Message)

HMAC-SHA3-512 for TCP-AO has the following values:

* MAC_alg: HMAC-SHA3-512.

* Traffic_Key: Variable; the result of the KDF.

* Message: The message to be authenticated, as specified in [RFC5925], Section 5.1.

4. Security Considerations

This document inherits all of the security considerations of the TCP-AO [RFC5925].

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
SHA256	This Document
SHA384	This Document
SHA512	This Document
SHA3-256	This Document
SHA3-384	This Document
SHA3-512	This Document

Table 1: IANA Actions

6. Acknowledgements

Thanks to Lars Eggert, Gorry 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

[DOI.10.6028_NIST.FIPS.180-4]
"Secure hash standard", National Institute of Standards and Technology (U.S.), DOI 10.6028/nist.fips.180-4, 2015, <https://doi.org/10.6028/nist.fips.180-4>.

[DOI.10.6028_NIST.FIPS.197]
"Advanced Encryption Standard (AES)", National Institute of Standards and Technology (U.S.), DOI 10.6028/nist.fips.197, 2001, <https://doi.org/10.6028/nist.fips.197>.

[DOI.10.6028_NIST.FIPS.202]
"SHA-3 standard :: permutation-based hash and extendable-output functions", National Institute of Standards and Technology (U.S.), DOI 10.6028/nist.fips.202, 2015, <https://doi.org/10.6028/nist.fips.202>.

[DOI.10.6028_NIST.SP.800-38B]
Dworkin, M., "Recommendation for block cipher modes of operation :: the CMAC mode for authentication", National Institute of Standards and Technology, DOI 10.6028/nist.sp.800-38b, 2016, <https://doi.org/10.6028/nist.sp.800-38b>.

[I-D.bonica-tcpm-extended-options]
Bonica, R. and T. Li, "TCP Extended Options", Work in Progress, Internet-Draft, draft-bonica-tcpm-extended-options-03, 22 March 2026, <https://datatracker.ietf.org/doc/html/draft-bonica-tcpm-extended-options-03>.

[RFC2104] Krawczyk, H., Bellare, M., and R. Canetti, "HMAC: Keyed-Hashing for Message Authentication", RFC 2104, DOI 10.17487/RFC2104, February 1997, <https://www.rfc-editor.org/rfc/rfc2104>.

[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>.

[RFC4615] Song, J., Poovendran, R., Lee, J., and T. Iwata, "The Advanced Encryption Standard-Cipher-based Message Authentication Code-Pseudo-Random Function-128 (AES-CMAC-PRF-128) Algorithm for the Internet Key Exchange Protocol (IKE)", RFC 4615, DOI 10.17487/RFC4615, August 2006, <https://www.rfc-editor.org/rfc/rfc4615>.

[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>.

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