Network Working Group J. Schaad
Internet-Draft August Cellars
Intended status: Informational 11 March 2019
Expires: 12 September 2019

CBOR Object Signing and Encryption (COSE): Hash Algorithms
draft-ietf-cose-hash-algs-00

Abstract

The CBOR Object Signing and Encryption (COSE) syntax [I-D.ietf-cose-rfc8152bis-struct] does not define any direct methods for using hash algorithms. There are however circumstances where hash algorithms are used: Indirect signatures where the hash of one or more contents are signed. X.509 certificate or other object identification by the use of a thumbprint. This document defines a set of hash algorithms that are identified by COSE Algorithm Identifiers.

Contributing to this document

The source for this draft is being maintained in GitHub. Suggested changes should be submitted as pull requests at TBD. Editorial changes can be managed in GitHub, but any substantial issues need to be discussed on the COSE mailing list.

Table of Contents

1. Introduction
    1.1. Requirements Terminology
    1.2. Open Issues
2. Hash Algorithm Identifiers
    2.1. SHA-1 Hash Algorithm
    2.2. SHA-2 Hash Algorithms
    2.3. SHAKE Algorithms
3. IANA Considerations
    3.1. COSE Algorithm Registry
4. Security Considerations
5. Normative References
6. Informative References
Author's Address

1. Introduction

The CBOR Object Signing and Encryption (COSE) syntax does not define any direct methods for the use of hash algorithms. It also does not define a structure syntax that is used to encode a digested object structure along the lines of the DigestedData ASN.1 structure in [CMS]. This omission was intentional as a structure consisting of jut a digest identifier, the content, and a digest value does not by itself provide any strong security service. Additional, an application is going to be better off defining this type of structure so that it can add any additional data that needs to be hashed as well as methods of obtaining the data.

While the above is true, there are some cases where having some standard hash algorithms defined for COSE with a common identifier makes a great deal of sense. Two of the cases where these are going to be used are:

* Indirect signing of content, and

* Object identification.

Indirect signing of content is a paradigm where the content is not directly signed, but instead a hash of the content is computed and that hash value, along with the hash algorithm, is included in the content that will be signed. Doing indirect signing allows for the a signature to be validated without first downloading all of the content associated with the signature. This capability can be of even grater importance in a constrained environment as not all of the content signed may be needed by the device.

The use of hashes to identify objects is something that has been very common. One of the primary things that has been identified by a hash function for secure message is a certificate. Two examples of this can be found in [ESS] and the newly defined COSE equivalents in [I-D.ietf-cose-x509].

1.1. Requirements Terminology

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 BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

1.2. Open Issues

* Are there additional SHA-2 formulations that need to be added or should some of the ones in this document be removed?

* Should additional hash algorithms be added to the document?

* Review the Recommended column in all of the tables to make sure that the values are correct.

* Are there recommendations that should be provided on what range of identifiers should be used for these algorithms? Inputs would include the expected frequency of use for each algorithm.

2. Hash Algorithm Identifiers

2.1. SHA-1 Hash Algorithm

The SHA-1 hash algorithm [RFC3174] wsa designed by the United States National Security Agenciy and published in 1995. Since that time a large amount of cryptographic analysis has been applied to this algorithm and a successful collision attack has been created ([SHA-1-collision]). The IETF formally started discouraging the use of SHA-1 with the publishing of [RFC6194].

Dispite the above, there are still times where SHA-1 needs to be used and therefore it makes sense to assign a point for the use of this hash algorithm. Some of these situations are with historic HSMs where only SHA-1 is implemented or where the SHA-1 value is used for the purpose of filtering and thus the collision resistance property is not needed.

Because of the known issues for SHA-1 and the fact that is should no longer be used, the algorithm will be registered with the recommendation of "Depreciated".

Name	Value	Description	Reference	Recommended
SHA-1	TBD6	SHA-1 Hash	[This Document]	Depreciated

Table 1: SHA-1 Hash Algorithm

2.2. SHA-2 Hash Algorithms

The family of SHA-2 hash algorithms [FIPS-180-4] was designed by the United States National Security Agency and published in 2001. Since that time some additional algorithms have been added to the original set to deal with length extension attacks and some performance issues. While the SHA-3 hash algorithms has been published since that time, the SHA-2 algorithms are still broadly used.

There are a number of different parameters for the SHA-2 hash functions. The set of hash functions which have been chosen for inclusion in this document are based on those different parameters and some of the trade-offs involved.

* *SHA-256/64* provides a truncated hash. The length of the truncation is designed to allow for smaller transmission size. The trade-off is that the odds that a collision will occur increase proportionally. Locations that use this hash function need either to analysis the potential problems with having a collision occur, or where the only function of the hash is to narrow the possible choices.

  The latter is the case for [I-D.ietf-cose-x509], the hash value is used to select possible certificates and, if there are multiple choices then, each choice can be tested by using the public key.

* *SHA-256* is probably the most common hash function used currently. SHA-256 is the most efficient hash algorithm for 32-bit hardware.

* *SHA-384* and *SHA-512* hash functions are more efficient when run on 64-bit hardware.

* *SHA-512/256* provides a hash function that runs more efficiently on 64-bit hardware, but offers the same security levels as SHA-256.

Name	Value	Description	Reference	Recommended
SHA-256/64	TBD1	SHA-2 256-bit Hash truncated to 64-bits	[This Document]	No

SHA-256	TBD2	SHA-2 256-bit Hash	[This Document]	Yes
SHA-384	TBD3	SHA-2 384-bit Hash	[This Document]	Yes
SHA-512	TBD4	SHA-2 512-bit Hash	[This Document]	Yes
SHA-512/256	TBD5	SHA-2 512-bit Hash truncated to 256-bits	[This Document]	Yes

Table 2: SHA-2 Hash Algorithms

2.3. SHAKE Algorithms

The family SHA-3 hash algorithms [FIPS-180-4] was the result of a competition run by NIST. The pair of algorithms known as SHAKE-128 and SHAKE-256 are the instances of SHA-3 that are currently being standardized in the IETF.

The SHA-3 hash algorithms have a significantly different structure than the SHA-3 hash algorithms. One of the benefits of this differences is that when computing a truncated SHAKE hash value, the value is not a prefix of a longer version of the same value.

MAYBE TEXT: Might not need to define truncated versions of this hash algorithm because the length of the resulting value is always going to generate a unique value since you cannot just truncate it like you can with SHA-1 and SHA-2.

Name	Value	Description	Reference	Recommended
SHAKE128	TBD10	128-bit SHAKE	[This Document]	Yes
SHAKE256	TBD11	256-bit SHAKE	[This Document]	Yes

Table 3: SHAKE Hash Functions

3. IANA Considerations

3.1. COSE Algorithm Registry

IANA is requested to register the following algorithms in the "COSE Algorithms" registry.

* The SHA-1 hash function found in Table 1.

* The set of SHA-2 hash functions found in Table 2.

* The set of SHAKE hash functions found in Table 3.

Many of the hash values produced are relatively long and as such the use of a two byte algorithm identifier seems reasonable. SHA-1 is tagged as deprecated and thus a longer algorithm identifier is appropriate even though it is a shorter hash value.

4. Security Considerations

There are security considerations:

5. Normative References

[FIPS-180-4]
National Institute of Standards and Technology, "Secure Hash Standard", FIPS PUB 180-4, August 2015.

[I-D.ietf-cose-rfc8152bis-struct]
Schaad, J., "CBOR Object Signing and Encryption (COSE) - Structures and Process", draft-ietf-cose-rfc8152bis-struct-01 (work in progress), 14 February 2019, <https://www.ietf.org/archive/id/draft-ietf-cose-rfc8152bis-struct-01>.

[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/info/rfc2119>.

[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/info/rfc8174>.

6. Informative References

[CMS] Housley, R., "Cryptographic Message Syntax (CMS)", STD 70, RFC 5652, DOI 10.17487/RFC5652, September 2009, <https://www.rfc-editor.org/info/rfc5652>.

[ESS] Hoffman, P., Ed., "Enhanced Security Services for S/MIME", RFC 2634, DOI 10.17487/RFC2634, June 1999, <https://www.rfc-editor.org/info/rfc2634>.

[I-D.ietf-cose-x509]
Schaad, J., "CBOR Object Signing and Encryption (COSE): Headers for carrying and referencing X.509 certificates", draft-ietf-cose-x509-00 (work in progress), 29 January 2019, <https://www.ietf.org/archive/id/draft-ietf-cose-x509-00>.

[RFC3174] Eastlake 3rd, D. and P. Jones, "US Secure Hash Algorithm 1 (SHA1)", RFC 3174, DOI 10.17487/RFC3174, September 2001, <https://www.rfc-editor.org/info/rfc3174>.

[RFC6194] Polk, T., Chen, L., Turner, S., and P. Hoffman, "Security Considerations for the SHA-0 and SHA-1 Message-Digest Algorithms", RFC 6194, DOI 10.17487/RFC6194, March 2011, <https://www.rfc-editor.org/info/rfc6194>.

[SHA-1-collision] Stevens, M., Bursztein, E., Karpman, P., Albertini, A., and Y. Markov, "The first collision for full SHA-1", February 2017, <https://shattered.io/static/shattered.pdf>.

Author's Address

Jim Schaad
August Cellars

Email: ietf@augustcellars.com