Network Working Group
Internet-Draft
Intended status: Informational
Expires: January 3, 2019

J. Schaad
August Cellars

July 2, 2018

CBOR Object Signing and Encryption (COSE): Headers for carrying and referencing X.509 certificates
draft-schaad-cose-x509-02

Abstract

This document defines a set of headers to identify and transport X.509 certificates in the CBOR Encoded Message (COSE) syntax. The document additionally defines a set of digest algorithms that are used in identifying certificates, as well as being available for other uses.

Contributing to this document

The source for this draft is being maintained in GitHub. Suggested changes should be submitted as pull requests at <https://github.com/cose-wg/X509>. Instructions are on that page as well. Editorial changes can be managed in GitHub, but any substantial issues need to be discussed on the COSE mailing list.

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 January 3, 2019.

Table of Contents

1. Introduction

In the process of writing [RFC8152] discussions where held on the question of X.509 certificates [RFC5280] and if there were needed. At the time there were no use cases presented that appeared to have a sufficient set of support to include these headers. Since that time a number of cases where X.509 certificate support is necessary have been defined. This document provides a set of headers that will allow applications to transport and refer to X.509 certificates in a consistent manner.

Some of the constrainted device situations are being used where an X.509 PKI is already installed. One of these situations is the 6tish environment for enrollment of devices where the certificates are installed at the factory. The [I-D.selander-ace-cose-ecdhe] draft was also written with the idea that long term certificates could be used to provide for authentication of devices and uses them to establish session keys. A final scenario is the use of COSE as a messaging application where long term existence of keys can be used along with a central authentication authority. The use of

certificates in this scenario allows for key managment to be used which is well understood.

Additionally, there has been an increasing need to have a set of standardized set of identifies for digest algorithms. Many cases one needs to sign a manifest which contains a pointer to a data structure, a digest algorithm and the digest value. This structure means that one is not required to include a document in order to have it correctly identified. As digest algoithms are also used in identification of certificates, an initial set of digest algorithms is defined in this document.

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.

2. X.509 COSE Headers

The use of X.509 certificates allows for an existing trust infrastructure to be used with COSE. This includes the full suite of enrollment protocols, trust anchors, trust chaining and revocation checking that have been defined over time by the IETF and other organizations. The key structures that have been defined in COSE currently do not support all of these properties although some may be found in COSE Web Tokens (CWT) [I-D.ietf-ace-cbor-web-token].

It is not necessarily expected that constrainted devices will fully support the evalaluation and processing of X.509 certificates, it is perfectly reasonable for a certificate to be assigned to a device which it can then provide to a relying party along with a signature or encrypted message, the relying party not being a constrained device.

Certificates obtained from any of these methods MUST still be validated. This validation can be done via the PKIX rules in [RFC5280] or by using a different trust structure, such as a trusted certificate distributer for self-signed certificates. The PKIX validation includes matching against the trust anchors configured for the application. These rules apply to certificates of a chain length of one as well as longer chains. If the application cannot establish a trust in the certificate, then it cannot be used.

The header parameters defined in this document are:

x5bag: This header parameters contains a bag of X.509 certificates. The set of certificates in this header are unordered and may contain self-signed certificates. The certificate bag can contain certificates which are completely extraneous to the message. An example of this would be to carry a certificate with a key agreement key usage in a signed message. As the certificates are unordered, the party evaluating the signature will need to do the necessary path building. Certificates needed for any particular chain to be built may be absent from the bag.

As this header element does not provide any trust, the header parameter can be in either a protected or unprotected header bag.

This header parameter allows for a single or a bag of X.509 certificates to be carried in the message.

* If a single certificate is conveyed, it is placed in a CBOR bstr.

* If multiple certificates are conveyed, a CBOR array of bstrs is used. Each certificate being in it's own slot.

x5chain: This header parameter contains an ordered array of X.509 certificates. The certificates are to be ordered starting with the certificate containing the end-entity key followed by the certificate which signed it and so on. The chain of certificates can be truncated if there is reason to believe that the relying party will already have it.

As this header element does not provide any trust, the header parameter can be in either a protected or unprotected header bag.

This header parameter allows for a single or a bag of X.509 certificates to be carried in the message.

* If a single certificate is conveyed, it is placed in a CBOR bstr.

* If multiple certificates are conveyed, a CBOR array of bstr is used. Each certificate being in it's own slot.

x5t: This header parameter provides the ability to identify an X.509 certificate by a hash value. The parameter is an array of two elements. The first element is an algorithm identifier which is a signed integer or a string containing the hash algorithm identifier. The second element is a binary string containing the hash value.

As this header element does not provide any trust, the header parameter can be in either a protected or unprotected header bag. For interoperability, applications which use this header parameter MUST support the hash algorithm 'sha256', but can use other hash algorithms.

x5u: This header parameter provides the ability to identify an X.509 certificate by a URL. The referenced resource can be any of the following media types:

* application/pkix-cert [RFC2585]

* application/pkcs7-mime; smime-type="certs-only" [I-D.ietf-lamps-rfc5751-bis]

* application/x-pem-file [RFC7468]

As this header element implies a trust relationship, the header parameter MUST be in the protected header bag.
The URL provided MUST provide integrity protection. For example, an HTTP or CoAP GET request to retrieve a certificate MUST use TLS [RFC5246] or DTLS. If the certificate does not chain to an existing trust anchor, the identity of the server MUST be configured as trusted to provide new trust anchors. This will normally be the situation when self-signed certificates are used.

The header paramters used in the following locations:

o COSE_Signature and COSE_Sign0 objects, in these objects they identify the key that was used for generating signature.

o COSE_recipient object, in this object they identify the key used by the sender for static-static key agreement algorithms.

name	label	value type	description
x5bag	TBD4	COSE_X509	An unordered bag of X.509 certificates
x5chain	TBD3	COSE_X509	An ordered chain of X.509 certificates
x5t	TBD1	COSE_CertHash	Hash of an X.509 certificate
x5u	TBD2	tstr	URL pointing to an X.509 certificate

Table 1: X.509 COSE Headers

Below is an equivalent CDDL [I-D.ietf-cbor-cddl] description of the text above

```
COSE_X509 = bstr / [ *certs: bstr ]
COSE_CertHash = [ hashAlg: (int / tstr), hashValue: bstr ]
```

3. Hash Algorithm Identifiers

The core COSE document did have a need for a standalone hash algorithm, and thus did not define any. In this document, two hash algorithms are defined for use with the 'x5t' header parameter. Nothing restricts their use in other contexts.

3.1. SHA-2 256-bit Hash

The SHA-2 256-bit algorithm is defined in [SHA2]. Define an algorithm identifier for SHA-256.

3.2. SHA-2 256-bit Hash trucated to 64 bits

This hash function uses the SHA-2 256-bit hash function as in the previous section, however it truncates the result to 64-bits for transmission. The fact that it is a trucated hash means that there is now a high likelyhood that colisions will occur, thus this hash function cannot be used in situations where a unique items is required to be identified. Luckly for the case of identifying a certificate that is not a requirement, the only requirement is that the number of potential certificates (and thus keys) to be tried is reduced to a small number. (Hopefully that number is one, but it can not be assumed to be.) After the set of certificates has been filtered down, the public key in each certificate will need to be

tried for the operation in question. The certificate can be validated either before or after it has been checked as working. The trade-offs involved are:

- Certificate validation before using the key will imply that more network traffic may be required in order to fetch certificates and do revocation checking.

- Certificate validation after using the key means that bad keys can be used and, if not carefully checked, the result may be used prior to completing the certificate validation. Using unvalidated keys can expose the device to more timing and oracle attacks as the attacker would be able to see if the key operation succeeded or failed as no network traffic to validate the certificate would ensue.

4. IANA Considerations

4.1. COSE Header Parameter Registry

It is requested that IANA create four new entries in the "COSE Header Parameters" registry. The content of these entries is:

Name: x5bag
Label: TBD4
Value Type: bstr | [+bstr]
Value Registry: N/A
Description: X.509 certificate bag
Reference: [[This Document]]

Name: x5chain
Label: TBD3
Value Type: bstr | [+bstr]
Value Registry: N/A
Description: X.509 certificate chain
Reference: [[This Document]

Name: x5t
Label: TBD1
Value Type: COSE_CertHash
Value Registry: N/A
Description: X.509 certificate thumbprint
Reference: [[This Document]]

Name: x5u
Label: TBD2
Value Type: tstr
Value Registry: N/A
Description: URL pointing to an X.509 certificate
Reference: [[This Description]]

4.2. COSE Algorithm Registry

It is requested that IANA create two new entries in the "COSE Algorithms" registry. The content of these entries is:

Name: SHA256
Value: TBD5
Description: SHA-256 Digest
Reference: [[This Document]]
Recommended: Yes

Name: SHA256/64
Value: TBD6
Description: SHA-256 Digest truncated to 64-bits
Reference: [[This Document]]
Recommended: No

Note to designated expert: It may be reasonable to use a single byte entry for the truncated algorthm, but I think it should be in the two

byte range. There is no reason not to place the full SHA-256 algorithm in the three byte range, but I expect it to be in the 2 byte range.

5. Security Considerations

There are security considerations:

6. References

6.1. 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/info/rfc2119>.

[RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, DOI 10.17487/RFC5280, May 2008, <https://www.rfc-editor.org/info/rfc5280>.

[RFC8152] Schaad, J., "CBOR Object Signing and Encryption (COSE)", RFC 8152, DOI 10.17487/RFC8152, July 2017, <https://www.rfc-editor.org/info/rfc8152>.

[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>.

[SHA2] National Institute of Standards and Technology (NIST), "Secure Hash Standard", FIPS 180-4, August 2015.

6.2. Informative References

[I-D.ietf-ace-cbor-web-token] Jones, M., Wahlstroem, E., Erdtman, S., and H. Tschofenig, "CBOR Web Token (CWT)", draft-ietf-ace-cbor-web-token-15 (work in progress), March 2018.

[I-D.ietf-cbor-cddl] Birkholz, H., Vigano, C., and C. Bormann, "Concise data definition language (CDDL): a notational convention to express CBOR data structures", draft-ietf-cbor-cddl-02 (work in progress), February 2018.

[I-D.ietf-lamps-rfc5751-bis]
Schaad, J., Ramsdell, B., and S. Turner, "Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 4.0 Message Specification", draft-ietf-lamps-rfc5751-bis-10 (work in progress), June 2018.

[I-D.selander-ace-cose-ecdhe]
Selander, G., Mattsson, J., and F. Palombini, "Ephemeral Diffie-Hellman Over COSE (EDHOC)", draft-selander-ace-cose-ecdhe-08 (work in progress), March 2018.

[RFC2585] Housley, R. and P. Hoffman, "Internet X.509 Public Key Infrastructure Operational Protocols: FTP and HTTP", RFC 2585, DOI 10.17487/RFC2585, May 1999, <https://www.rfc-editor.org/info/rfc2585>.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, DOI 10.17487/RFC5246, August 2008, <https://www.rfc-editor.org/info/rfc5246>.

[RFC7468] Josefsson, S. and S. Leonard, "Textual Encodings of PKIX, PKCS, and CMS Structures", RFC 7468, DOI 10.17487/RFC7468, April 2015, <https://www.rfc-editor.org/info/rfc7468>.

[TRUNCATE]
National Institute of Standards and Technology (NIST), "Recommendation fro Applications Using Approved Hash Algorithms", FIPS 800-107, August 2012.

Author's Address

Jim Schaad
August Cellars

Email: ietf@augustcellars.com