

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

Abstract

The CBOR Signing And Encrypted Message (COSE) structure uses references to keys in general. For some algorithms, additional properties are defined which carry parts of keys as needed. The COSE Key structure is used for transporting keys outside of COSE messages. This document extends the way that keys can be identified and transported by providing attributes that refer to or contain X.509 certificates.

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 December 22, 2019.

Copyright Notice

Copyright (c) 2019 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 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.

Table of Contents

1.	Introduction	2
1.1.	Requirements Terminology	3
1.2.	Open Questions	3
2.	X.509 COSE Headers	3
3.	X.509 certificates and static-static ECDH	6
4.	IANA Considerations	7
4.1.	COSE Header Parameter Registry	7
4.2.	COSE Header Algorithm Parameter Registry	8
5.	Security Considerations	8
6.	References	8
6.1.	Normative References	8
6.2.	Informative References	9
	Author's Address	10

[1.](#) Introduction

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

Some of the constrained 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 management to be used which is well understood.

Example COSE messages for the various headers defined below can be found at <https://github.com/cose-wg/Examples>. THIS IS NOT YET DONE BUT SHOULD BE COMING NOT LONG AFTER THE F2F MEETING.

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 Questions

Should we define an extended key usage?

Revocation info?

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) [[RFC8392](#)].

It is not necessarily expected that constrained devices will fully support the evaluation 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 distributor 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 attributes defined in this document are:

x5bag: This header attributes 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 attribute can be in either a protected or unprotected header attribute.

This header attribute 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 attribute 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. There is no requirement for the entire chain to be present in the element if there is reason to believe that the relying party will already have it. This means that the relying party is still required to do path building, but that a candidate path is proposed in this attribute.

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

This header attribute allows for a single or a chain 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.

x5t: This header attribute provides the ability to identify an X.509 certificate by a hash value. The attribute is an array of two elements. The first element is an algorithm identifier which is an 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 attribute can be in either a protected or unprotected header attribute.

For interoperability, applications which use this header attribute MUST support the hash algorithm 'sha256', but can use other hash algorithms.

x5u: This header attribute 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/pem-certificate-chain [[I-D.ietf-acme-acme](#)]

As this header attribute implies a trust relationship, the attribute MUST be in the protected attributes.

The URL provided MUST provide integrity protection and server authentication. For example, an HTTP or CoAP GET request to retrieve a certificate MUST use TLS [[RFC8446](#)] or DTLS [[I-D.ietf-tls-dtls13](#)]. If the certificate does not chain to an existing trust anchor, the certificate MUST NOT be trusted unless the server is configured as trusted to provide new trust anchors. This will normally be the situation when self-signed certificates are used.

The header attributes are 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 objects, in this location they may be used to identify the certificate for the recipient of the message.

Name	Value	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	uri	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 / [ 2*certs: bstr ]
COSE_CertHash = [ hashAlg: (int / tstr), hashValue: bstr ]
```

3. X.509 certificates and static-static ECDH

The header attributes defined in the previous section are used to identify the recipient certificates for the ECDH key agreement algorithms. In this section we define the algorithm specific parameters that are used for identifying or transporting the senders key for static-static key agreement algorithms.

These attributes are defined analogously to those in the previous section. There is no definition for the certificate bag as the same attribute would be used for both the sender and recipient certificates.

x5chain-sender: This header attribute contains the chain of certificates starting with the sender's key exchange certificate. The structure is the same as 'x5bag'.

x5t-sender: This header attribute contains the hash value for the sender's key exchange certificate. The structure is the same as 'x5t'.

x5u-sender: This header attribute contains a URL for the sender's key exchange certificate. The structure and processing are the same as 'x5u'.

Name	Value	Type	Algorithm	Description
x5t-sender	TBD	COSE_CertHash	ECDH-SS+HKDF-256, ECDH-SS+HKDF-512, ECDH-SS+A128KW, ECDH-SS+AES192KW, ECDH-SS+AES256KW	Thumbprint for the senders X.509 certificate
x5u-sender	TBD	uri	ECDH-SS+HKDF-256, ECDH-SS+HKDF-512, ECDH-SS+A128KW, ECDH-SS+AES192KW, ECDH-SS+AES256KW	URL for the senders X.509 certificate
x5chain-sender	TBD	COSE_X509	ECDH-SS+HKDF-256, ECDH-SS+HKDF-512, ECDH-SS+A128KW, ECDH-SS+AES192KW, ECDH-SS+AES256KW	static key X.509 certificate chain

Table 2: Static ECDH Algorithm Values

4. IANA Considerations

4.1. COSE Header Parameter Registry

IANA is requested to register the new COSE Header items in Table 1 in the "COSE Header Parameters" registry.

4.2. COSE Header Algorithm Parameter Registry

IANA is requested to register the new COSE Header items in Table 2 in the "COSE Header Algorithm Parameters" registry.

5. Security Considerations

Establishing trust in a certificate is a vital part of processing. Trust cannot be assumed whenever a new self-signed certificate appears on the client, instead a well defined process is required. One common way for a new trust anchor to be added (or removed) from a device is by doing a new firmware upgrade.

In constrained systems, there is a trade-off between the order of checking the signature and checking the certificate for validity. Validating certificates can require that network resources be accessed in order to get revocation information or retrieve certificates during path building. Doing the network access can consume resources dealing with power and network bandwidth. On the other hand, an oracle can potentially be built based on if the network resources are only accessed if the signature validation passes. In any event, both the signature and certificate validation MUST be checked before acting on any requests.

As called out in the COSE algorithms document [[I-D.ietf-cose-rfc8152bis-algs](#)] basic checking on the keys in a certificate needs to be performed prior to using them. These can include validating that points are on curves for elliptical curve algorithms and that sizes of keys are acceptable for RSA. The use of unvalidated keys can lead either to loss of security or excessive consumption of resources.

6. References

6.1. Normative References

- [I-D.ietf-cose-rfc8152bis-struct]
Schaad, J., "CBOR CBOR Object Signing and Encryption (COSE): Structures and Process", [draft-ietf-cose-rfc8152bis-struct-02](#) (work in progress), March 2019.
- [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>>.
- [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>>.

6.2. Informative References

- [I-D.ietf-acme-acme]
Barnes, R., Hoffman-Andrews, J., McCarney, D., and J. Kasten, "Automatic Certificate Management Environment (ACME)", [draft-ietf-acme-acme-18](#) (work in progress), December 2018.
- [I-D.ietf-cbor-cddl]
Birkholz, H., Vigano, C., and C. Bormann, "Concise data definition language (CDDL): a notational convention to express CBOR and JSON data structures", [draft-ietf-cbor-cddl-08](#) (work in progress), March 2019.
- [I-D.ietf-cose-rfc8152bis-algs]
Schaad, J., "CBOR Object Signing and Encryption (COSE): Initial Algorithms", [draft-ietf-cose-rfc8152bis-algs-02](#) (work in progress), March 2019.
- [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-12](#) (work in progress), September 2018.
- [I-D.ietf-tls-dtls13]
Rescorla, E., Tschofenig, H., and N. Modadugu, "The Datagram Transport Layer Security (DTLS) Protocol Version 1.3", [draft-ietf-tls-dtls13-31](#) (work in progress), March 2019.
- [I-D.selander-ace-cose-ecdhe]
Selander, G., Mattsson, J., and F. Palombini, "Ephemeral Diffie-Hellman Over COSE (EDHOC)", [draft-selander-ace-cose-ecdhe-13](#) (work in progress), March 2019.

- [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>>.
- [RFC8152] Schaad, J., "CBOR Object Signing and Encryption (COSE)", [RFC 8152](#), DOI 10.17487/RFC8152, July 2017, <<https://www.rfc-editor.org/info/rfc8152>>.
- [RFC8392] Jones, M., Wahlstroem, E., Erdtman, S., and H. Tschofenig, "CBOR Web Token (CWT)", [RFC 8392](#), DOI 10.17487/RFC8392, May 2018, <<https://www.rfc-editor.org/info/rfc8392>>.
- [RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", [RFC 8446](#), DOI 10.17487/RFC8446, August 2018, <<https://www.rfc-editor.org/info/rfc8446>>.

Author's Address

Jim Schaad
August Cellars

Email: ietf@augustcellars.com

