

Network Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: 26 November 2022

J. Schaad  
August Cellars  
25 May 2022

CBOR Object Signing and Encryption (COSE): Header parameters for  
carrying and referencing X.509 certificates  
draft-ietf-cose-x509-09

Abstract

The CBOR Signing And Encrypted Message (COSE) structure uses references to keys in general. For some algorithms, additional properties are defined which carry parameters relating to 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

This note is to be removed before publishing as an RFC.

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 26 November 2022.

## Copyright Notice

Copyright (c) 2022 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
2. X.509 COSE Header Parameters . . . . .	3
3. X.509 certificates and static-static ECDH . . . . .	8
4. IANA Considerations . . . . .	9
4.1. COSE Header Parameter Registry . . . . .	9
4.2. COSE Header Algorithm Parameter Registry . . . . .	9
4.3. Media Type application/cose-x509 . . . . .	10
5. Security Considerations . . . . .	11
6. References . . . . .	12
6.1. Normative References . . . . .	12
6.2. Informative References . . . . .	13
Appendix A. Acknowledgements . . . . .	14
Author's Address . . . . .	14

## 1. Introduction

In the process of writing [RFC8152], the working group discussed X.509 certificates [RFC5280] and decided that no use cases were presented that showed a need to support certificates. Since that time, a number of cases have been defined in which X.509 certificate support is necessary, and by implication, applications will need a documented and consistent way to handle such certificates. This document defines a set of attributes that will allow applications to transport and refer to X.509 certificates in a consistent manner.

In some of these cases, a constrained device is being deployed in the context of an existing X.509 PKI: for example, [I-D.ietf-anima-constrained-voucher] describes a device enrollment solution that relies on the presence of a factory-installed certificate on the device. The [I-D.ietf-lake-edhoc] 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. Another possible scenario is the use of COSE as the basis for a secure messaging application. This scenario assumes the presence of long term keys and a central authentication authority. Basing such an application on public key certificates allows it to make use of well established key management disciplines.

### 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 Header Parameters

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 themselves will evaluate and process X.509 certificates: it is perfectly reasonable for a constrained device to be provisioned with a certificate that it subsequently provides to a relying party - along with a signature or encrypted message - on the assumption that the relying party is not a constrained device, and is capable of performing the required certificate evaluation and processing. It is also reasonable that a constrained device would have the hash of a certificate associated with a public key and be configured to use a public key for that thumbprint, but without performing the certificate evaluation or even having the entire certificate. In any case, there still needs to be an entity that is responsible for handling the possible certificate revocation.

Parties that intend to rely on the assertions made by a certificate obtained from any of these methods still need to validate it. This validation can be done according to 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 when the validation succeeds in a single step as well as when certificate chains need to be built. If the application cannot establish trust in the certificate, the public key contained in the certificate cannot be used for cryptographic operations.

The header parameters defined in this document are:

**x5bag:** This header parameter contains a bag of X.509 certificates. The set of certificates in this header parameter is unordered and may contain self-signed certificates. Note that there could be duplicate certificates. The certificate bag can contain certificates which are completely extraneous to the message. (An example of this would be where a signed message is being used to transport a certificate containing a key agreement key.) As the certificates are unordered, the party evaluating the signature will need to be capable of building the certificate path as necessary. That party will also have to take into account that the bag may not contain the full set of certificates needed to build any particular chain.

The trust mechanism MUST process any certificates in this parameter as untrusted input. The presence of a self-signed certificate in the parameter MUST NOT cause the update of the set of trust anchors without some out-of-band confirmation. As the contents of this header parameter are untrusted input, the header parameter can be in either the protected or unprotected header bucket. Sending the header parameter in the unprotected header bucket allows an intermediary to remove or add certificates.

The end-entity certificate MUST be integrity protected by COSE. This can e.g. be done by sending the header parameter in the protected header, sending a x5bag in the unprotected header combined with a x5t in the protected header, or including the end-entity certificate in the external\_aad.

This header parameter allows for a single X.509 certificate 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 byte string.

- \* If multiple certificates are conveyed, a CBOR array of byte strings is used, with each certificate being in its own byte string.

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. There is no requirement for the entire chain to be present in the element if there is reason to believe that the relying party already has, or can locate the missing certificates. This means that the relying party is still required to do path building, but that a candidate path is proposed in this header parameter.

The trust mechanism MUST process any certificates in this parameter as untrusted input. The presence of a self-signed certificate in the parameter MUST NOT cause the update of the set of trust anchors without some out-of-band confirmation. As the contents of this header parameter are untrusted input, the header parameter can be in either the protected or unprotected header bucket. Sending the header parameter in the unprotected header bucket allows an intermediary to remove or add certificates.

The end-entity certificate MUST be integrity protected by COSE. This can e.g. be done by sending the header parameter in the protected header, sending a x5chain in the unprotected header combined with a x5t in the protected header, or including the end-entity certificate in the external\_aad as.

This header parameter allows for a single X.509 certificate 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 byte string.
- \* If multiple certificates are conveyed, a CBOR array of byte strings is used, with each certificate being in its own byte string.

x5t: This header parameter identifies the end-entity X.509

certificate by a hash value (a thumbprint). The 'x5t' header parameter is represented as an array of two elements. The first element is an algorithm identifier which is an integer or a string containing the hash algorithm identifier corresponding to the Value column (integer or text string) of the algorithm registered in the "COSE Algorithms" registry <https://www.iana.org/assignments/cose/cose.xhtml#algorithms>. The second element is a binary string containing the hash value computed over the DER encoded certificate.

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

The identification of the end-entity certificate MUST be integrity protected by COSE. This can be done by sending the header parameter in the protected header or including the end-entity certificate in the external\_aad.

The 'x5t' header parameter can be used alone or together with the 'x5bag', 'x5chain', or 'x5u' header parameters to provide integrity protection of the end-entity certificate.

For interoperability, applications which use this header parameter MUST support the hash algorithm 'SHA-256', but can use other hash algorithms. This requirement allows for different implementations to be configured to use an interoperable algorithm, but does not preclude the use (by prior agreement) of other algorithms.

RFC Editor please remove the following two paragraphs:

During AD review, a question was raised about how effective the previous statement is in terms of dealing with a MTI algorithm. There needs to be some type of arrangement between the parties to agree that a specific hash algorithm is going to be used in computing the thumbprint. Making it a MUST use would make that true, but it then means that agility is going to be very difficult.

The worry is that while SHA-256 may be mandatory, if a sender supports SHA-256 but only sends SHA-512 then the recipient which only does SHA-256 would not be able to use the thumbprint. In that case both applications would conform to the specification, but still not be able to inter-operate.

x5u: This header parameter provides the ability to identify an X.509 certificate by a URI [RFC3986]. It contains a CBOR text string. The referenced resource can be any of the following media types:

- \* application/pkix-cert [RFC2585]
- \* application/pkcs7-mime; smime-type="certs-only" [RFC8551]
- \* application/cose-x509 Section 4.3
- \* application/cose-x509; usage=chain Section 4.3

When the application/cose-x509 media type is used, the data is a CBOR sequence of single-entry COSE\_X509 structures (encoding "bstr"). If the parameter "usage" is set to "chain", this sequence indicates a certificate chain.

The end-entity certificate MUST be integrity protected by COSE. This can e.g. be done by sending the x5u in the unprotected or protected header combined with a x5t in the protected header, or including the end-entity certificate in the external\_aad. As the end-entity certificate is integrity protected by COSE, the URI does not need to provide any protection.

If a retrieved certificate does not chain to an existing trust anchor, that certificate MUST NOT be trusted unless the URI provided integrity protection and server authentication and the server is configured as trusted to provide new trust anchors or if an out-of-band confirmation can be received for trusting the retrieved certificate. In case an HTTP or CoAP GET request is used to retrieve a certificate, TLS [RFC8446], DTLS [I-D.ietf-tls-dtls13] or OSCORE [RFC8613] SHOULD be used.

The header parameters are used in the following locations:

- \* COSE\_Signature and COSE\_Sign1 objects: in these objects they identify the certificate to be used for validating the signature.
- \* COSE\_recipient objects: in this location they identify the certificate for the recipient of the message.

The labels assigned to each header parameter can be found in the following table.

Name	Label	Value Type	Description
x5bag	32	COSE_X509	An unordered bag of X.509 certificates
x5chain	33	COSE_X509	An ordered chain of X.509 certificates
x5t	34	COSE_CertHash	Hash of an X.509 certificate
x5u	35	uri	URI pointing to an X.509 certificate

Table 1: X.509 COSE Header Parameters

Below is an equivalent CDDL [RFC8610] description of the text above.

```
COSE_X509 = bstr / [ 2*certs: bstr ]
COSE_CertHash = [ hashAlg: (int / tstr), hashValue: bstr ]
```

The content of the bstr are the bytes of a DER encoded certificate.

### 3. X.509 certificates and static-static ECDH

The header parameters 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 sender's 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 parameter contains the chain of certificates starting with the sender's key exchange certificate. The structure is the same as 'x5chain'.

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

**x5u-sender:** This header parameter contains a URI for the sender's



key exchange certificate. The structure and processing are the same as 'x5u'.

Name	Label	Type	Algorithm	Description
x5t-sender	TBD	COSE_CertHash	ECDH-SS+HKDF-256, ECDH-SS+HKDF-512, ECDH-SS+A128KW, ECDH-SS+A192KW, ECDH-SS+A256KW	Thumbprint for the sender's X.509 certificate
x5u-sender	TBD	uri	ECDH-SS+HKDF-256, ECDH-SS+HKDF-512, ECDH-SS+A128KW, ECDH-SS+A192KW, ECDH-SS+A256KW	URI for the sender's X.509 certificate
x5chain-sender	TBD	COSE_X509	ECDH-SS+HKDF-256, ECDH-SS+HKDF-512, ECDH-SS+A128KW, ECDH-SS+A192KW, ECDH-SS+A256KW	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 parameters in Table 1 in the "COSE Header Parameters" registry. The "Value Registry" field is empty for all of the items. For each item, the 'Reference' field points to this document.

### 4.2. COSE Header Algorithm Parameter Registry

IANA is requested to register the new COSE Header Algorithm parameters in Table 2 in the "COSE Header Algorithm Parameters" registry. For each item, the 'Reference' field points to this document.

#### 4.3. Media Type application/cose-x509

When the application/cose-x509 media type is used, the data is a CBOR sequence of single-entry COSE\_X509 structures (encoding "bstr"). If the parameter "usage" is set to "chain", this sequence indicates a certificate chain.

IANA is requested to register the following media type [RFC6838]:

Type name: application

Subtype name: cose-x509

Required parameters: N/A

Optional parameters: usage

- \* Can be absent to provide no further information about the intended meaning of the order in the CBOR sequence of certificates.
- \* Can be set to "chain" to indicate that the sequence of data items is to be interpreted as a certificate chain.

Encoding considerations: binary

Security considerations: See the Security Considerations section of RFCthis.

Interoperability considerations: N/A

Published specification: RFCthis

Applications that use this media type: Applications that employ COSE and use X.509 as a certificate type.

Fragment identifier considerations: N/A

Additional information: Deprecated alias names for this type: N/A

Magic number(s): N/A

File extension(s): N/A

Macintosh file type code(s): N/A

Person & email address to contact for further information:  
iesg@ietf.org

Intended usage: COMMON

Restrictions on usage: N/A

Author: COSE WG

Change controller: IESG

Provisional registration? (standards tree only): no

## 5. Security Considerations

Establishing trust in a certificate is a vital part of processing. A major component of establishing trust is determining what the set of trust anchors are for the process. A new self-signed certificate appearing on the client cannot be a trigger to modify the set of trust anchors, because a well-defined trust-establishment 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. The resulting network access can consume power and network bandwidth. On the other hand, if the certificates are validated after the signature is validated, an oracle can potentially be built based on detecting the network resources which is only done if the signature validation passes. In any event, both the signature and certificate validation MUST be completed successfully before acting on any requests.

Unless it is known that the CA required proof-of-possession of the subject's private key to issue an end-entity certificate, the end-entity certificate MUST be integrity protected by COSE. Without proof-of-possession, an attacker can trick the CA to issue an identity-misbinding certificate with someone else's "borrowed" public-key but with a different subject. A MITM attacker can then perform an identity-misbinding attack by replacing the real end-entity certificate in COSE with such an identity-misbinding certificate.

End-entity X.509 certificates contain identities that a passive on-path attacker eavesdropping on the conversation can use to identify and track the subject. COSE does not provide identity protection by itself and the x5t and x5u header parameters are just alternative permanent identifiers and can also be used to track the subject. To provide identity protection, COSE can be sent inside another security protocol providing confidentiality.

Before using the key in a certificate, the key MUST be checked against the algorithm to be used and any algorithm specific checks need to be made. These checks can include validating that points are on curves for elliptical curve algorithms, and that sizes of RSA keys are of an acceptable size. The use of unvalidated keys can lead either to loss of security or excessive consumption of resources (for example using a 200K RSA key).

When processing the x5u header parameter the security considerations of [RFC3986] and specifically those defined in Section 7.1 of [RFC3986] also apply.

Regardless of the source, certification path validation is an important part of establishing trust in a certificate. Section 6 of [RFC5280] provides guidance for the path validation. The security considerations of [RFC5280] are also important for the correct usage of this document.

Protecting the integrity of the x5bag, x5chain and x5t contents by placing them in the protected header bucket can help mitigate some risks of a misbehaving certificate authority (cf. Section 5.1 of [RFC2634]).

The security of the algorithm used for 'x5t' does not affect the security of the system as this header parameter selects which certificate that is already present on the system should be used, but it does not provide any trust.

## 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>>.
- [RFC8949] Bormann, C. and P. Hoffman, "Concise Binary Object Representation (CBOR)", STD 94, RFC 8949, DOI 10.17487/RFC8949, December 2020, <<https://www.rfc-editor.org/info/rfc8949>>.

## 6.2. Informative References

- [I-D.ietf-anima-constrained-voucher]  
Richardson, M., Stok, P. V. D., Kampanakis, P., and E. Dijk, "Constrained Bootstrapping Remote Secure Key Infrastructure (BRSKI)", Work in Progress, Internet-Draft, draft-ietf-anima-constrained-voucher-17, 7 April 2022, <<https://www.ietf.org/archive/id/draft-ietf-anima-constrained-voucher-17.txt>>.
- [I-D.ietf-lake-edhoc]  
Selandier, G., Mattsson, J. P., and F. Palombini, "Ephemeral Diffie-Hellman Over COSE (EDHOC)", Work in Progress, Internet-Draft, draft-ietf-lake-edhoc-14, 18 May 2022, <<https://www.ietf.org/archive/id/draft-ietf-lake-edhoc-14.txt>>.
- [I-D.ietf-tls-dtls13]  
Rescorla, E., Tschofenig, H., and N. Modadugu, "The Datagram Transport Layer Security (DTLS) Protocol Version 1.3", Work in Progress, Internet-Draft, draft-ietf-tls-dtls13-43, 30 April 2021, <<https://www.ietf.org/archive/id/draft-ietf-tls-dtls13-43.txt>>.
- [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>>.

- [RFC2634] Hoffman, P., Ed., "Enhanced Security Services for S/MIME", RFC 2634, DOI 10.17487/RFC2634, June 1999, <<https://www.rfc-editor.org/info/rfc2634>>.
- [RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, DOI 10.17487/RFC3986, January 2005, <<https://www.rfc-editor.org/info/rfc3986>>.
- [RFC6838] Freed, N., Klensin, J., and T. Hansen, "Media Type Specifications and Registration Procedures", BCP 13, RFC 6838, DOI 10.17487/RFC6838, January 2013, <<https://www.rfc-editor.org/info/rfc6838>>.
- [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>>.
- [RFC8551] Schaad, J., Ramsdell, B., and S. Turner, "Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 4.0 Message Specification", RFC 8551, DOI 10.17487/RFC8551, April 2019, <<https://www.rfc-editor.org/info/rfc8551>>.
- [RFC8610] Birkholz, H., Vigano, C., and C. Bormann, "Concise Data Definition Language (CDDL): A Notational Convention to Express Concise Binary Object Representation (CBOR) and JSON Data Structures", RFC 8610, DOI 10.17487/RFC8610, June 2019, <<https://www.rfc-editor.org/info/rfc8610>>.
- [RFC8613] Selander, G., Mattsson, J., Palombini, F., and L. Seitz, "Object Security for Constrained RESTful Environments (OSCORE)", RFC 8613, DOI 10.17487/RFC8613, July 2019, <<https://www.rfc-editor.org/info/rfc8613>>.

## Appendix A. Acknowledgements

### Author's Address

Jim Schaad  
August Cellars

Email: [ietf@augustcellars.com](mailto:ietf@augustcellars.com)