

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: May 19, 2021

S. Raza
J. Hoeglund
RISE AB
G. Selander
J. Mattsson
Ericsson AB
M. Furuher
Nexus Group
November 15, 2020

CBOR Encoding of X.509 Certificates (CBOR Certificates)
draft-mattsson-cose-cbor-cert-compress-03

Abstract

This document specifies a CBOR encoding of PKIX profiled X.509 Certificates. The resulting certificates are called "CBOR certificates". The CBOR encoding supports a large subset of [RFC 5280](#), while at the same time producing very small sizes for certificates compatible with [RFC 7925](#). The CBOR encoding can be used to compress DER encoded X.509 certificates to encode natively signed certificates. When used to compress DER encoded X.509 certificates, the CBOR encoding can in many cases compress [RFC 7925](#) profiled certificates with over 50%. The document also specifies COSE headers for CBOR certificates as well as a TLS certificate type for CBOR certificates.

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 May 19, 2021.

Copyright Notice

Copyright (c) 2020 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	3
2.	Notational Conventions	4
3.	CBOR Encoding	4
3.1.	Message Fields	5
3.2.	Encoding of Extensions	9
4.	Compliance Requirements for Constrained IoT	10
5.	Deployment settings	10
6.	Expected Certificate Sizes	11
7.	Natively Signed CBOR Certificates	11
8.	Security Considerations	12
9.	IANA Considerations	12
9.1.	CBOR Certificate Types Registry	12
9.2.	CBOR Attribute Type Registry	13
9.3.	CBOR Extension Type Registry	13
9.4.	CBOR Extended Key Usage Registry	14
9.5.	CBOR Subject Alternative Name Registry	15
9.6.	CBOR Certificate Signature Algorithms Registry	15
9.7.	CBOR Certificate Public Key Algorithms Registry	16
9.8.	COSE Header Parameters Registry	17
9.9.	TLS Certificate Types Registry	17
10.	References	18
10.1.	Normative References	18
10.2.	Informative References	19
Appendix A.	Example CBOR Certificates	20
A.1.	Example RFC 7925 profiled X.509 Certificate	20
A.2.	Example HTTPS X.509 Certificate	22
Appendix B.	X.509 Certificate Profile, ASN.1	25
	Acknowledgments	27
	Authors' Addresses	27

1. Introduction

One of the challenges with deploying a Public Key Infrastructure (PKI) for the Internet of Things (IoT) is the size and encoding of X.509 public key certificates [RFC5280], since those are not optimized for constrained environments [RFC7228]. More compact certificate representations are desirable. Due to the current PKI usage of DER encoded X.509 certificates, keeping compatibility with DER encoded X.509 is necessary at least for a transition period. However, the use of a more compact encoding with the Concise Binary Object Representation (CBOR) [RFC7049] reduces the certificate size significantly which has known performance benefits in terms of decreased communication overhead, power consumption, latency, storage, etc.

CBOR is a data format designed for small code size and small message size. CBOR builds on the JSON data model but extends it by e.g. encoding binary data directly without base64 conversion. In addition to the binary CBOR encoding, CBOR also has a diagnostic notation that is readable and editable by humans. The Concise Data Definition Language (CDDL) [RFC8610] provides a way to express structures for protocol messages and APIs that use CBOR. [RFC8610] also extends the diagnostic notation.

CBOR data items are encoded to or decoded from byte strings using a type-length-value encoding scheme, where the three highest order bits of the initial byte contain information about the major type. CBOR supports several different types of data items, in addition to integers (int, uint), simple values (e.g. null), byte strings (bstr), and text strings (tstr), CBOR also supports arrays [] of data items, maps {} of pairs of data items, and sequences of data items. For a complete specification and examples, see [RFC7049], [RFC8610], and [RFC8742].

RFC 7925 [RFC7925] specifies a certificate profile for Internet of Things deployments which can be applied for lightweight certificate based authentication with e.g. TLS [RFC8446], DTLS [I-D.ietf-tls-dtls13], COSE [RFC8152], or EDHOC [I-D.ietf-lake-edhoc]. This document specifies a CBOR encoding which can support large parts of [RFC5280] based on [X.509-IoT]. The encoding support all [RFC7925] profiled X.509 certificates. Two variants are defined using the same CBOR encoding and differing only in what is being signed:

- o CBOR compression of DER encoded X.509 certificates [RFC5280], which can be decompressed into the original DER encoded X.509 certificate.

- o Natively signed CBOR certificates, which further optimizes the performance in constrained environments but is not backwards compatible with [\[RFC5280\]](#), see [Section 7](#).

This document specifies COSE headers for use of the CBOR certificates with COSE, see [Section 9.8](#). The document also specifies a TLS certificate type for use of the CBOR certificates with TLS (with or without additional TLS certificate compression), see [Section 9.9](#).

2. Notational Conventions

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.

This specification makes use of the terminology in [\[RFC5280\]](#), [\[RFC7049\]](#), [\[RFC7228\]](#), and [\[RFC8610\]](#).

3. CBOR Encoding

This section specifies the content and encoding for CBOR certificates, with the overall objective to produce a very compact representation supporting large parts of [\[RFC5280\]](#) and everything in [\[RFC7925\]](#). In the CBOR encoding, static fields are elided, elliptic curve points are compressed, OID are replaced with short integers, time values are compressed, and redundant encoding is removed. Combining these different components reduces the certificate size significantly, which is not possible with general purpose compressions algorithms, see Figure 1.

The CBOR certificate can be either a CBOR compressed X.509 certificate, in which case the signature is calculated on the DER encoded ASN.1 data in the X.509 certificate, or a natively signed CBOR certificate, in which case the signature is calculated directly on the CBOR encoded data (see [Section 7](#)). In both cases the certificate content is adhering to the restrictions given by [\[RFC5280\]](#). When used as for compression of an existing X.509 certificate, the encoding only works on canonical encoded certificates. The encoding is known to work with DER but might work with other canonical encodings. The compression does not work for BER encoded certificates.

In the encoding described below the order of elements in arrays are always encoded in the same order as the elements or the corresponding SEQUENCE or SET in the DER encoding.

3.1. Message Fields

The X.509 fields and their CBOR encodings are listed below.

CBOR certificates are defined in terms of DER encoded [\[RFC5280\]](#) X.509 certificates:

- o version. The 'version' field is known (fixed to v3) and is omitted in the CBOR encoding.
- o serialNumber. The 'serialNumber' INTEGER value field is encoded as a CBOR byte string 'certificateSerialNumber'. Any leading 0x00 byte (to indicate that the number is not negative) is omitted.
- o signatureAlgorithm. The 'signatureAlgorithm' field is encoded as a CBOR int 'issuerSignatureAlgorithm' (see [Section 9.6](#)) or a relativeOID byte string. Algorithms with parameters are not supported except RSA algorithms that use parameters = NULL.
- o signature. The 'signature' field is always the same as the 'signatureAlgorithm' field and always omitted from the CBOR encoding.
- o issuer. In the general case, the sequence of 'RelativeDistinguishedName' is encoded as CBOR array of CBOR arrays of Attributes, where each Attribute type and value is encoded as a (CBOR int, CBOR text string) pair. Each AttributeType is encoded as a CBOR int (see Figure 3), where the sign is used to represent the character string type; positive for printableString, negative for utf8String. The string types teletexString, universalString, and bmpString are not supported. If exactly one 'RelativeDistinguishedName' is present, the outer array is omitted, and issuer is encoded as a single CBOR array. If a RelativeDistinguishedName contains a single Attribute containing an utf8String encoded 'common name', the int is omitted and the Attribute is encoded as a single CBOR text string. If the utf8String encoded 'common name' contains an EUI-64 mapped from a 48-bit MAC address (i.e. of the form "hh-hh-hh-ff-fe-hh-hh-hh) it is encoded as a CBOR byte string of length 6. Other EUI-64 is encoded as a CBOR byte string of length 8.
- o validity. The 'notBefore' and 'notAfter' fields are ASCII string of the form "yyymmddHHMMSSZ" for UTCTime and "yyyymmddHHMMSSZ" for GeneralizedTime. They are encoded as unsigned integers using the following invertible encoding (Horner's method with different bases).

$$n = SS + 61 * (MM + 60 * (HH + 24 * (dd + 32 * (mm + 13 * (yy)yy))))$$

They are encoded as a byte string, which is interpreted as an unsigned integer n in network byte order. UTCTime and GeneralizedTime are encoded as a byte strings of length 4 and 5 respectively. Decoding can be done by a succession of modulo and subtraction operations. I.e. $SS = n \bmod 61$, $MM = ((n - SS) / 61) \bmod 60$, etc.

- o **subject.** The 'subject' is encoded exactly like issuer.
- o **subjectPublicKeyInfo.** The 'algorithm' field is encoded as the CBOR int 'subjectPublicKeyAlgorithm' (see [Section 9.7](#)) or a relativeOID byte string. Algorithms with parameters are not supported except id-ecPublicKey with named curves and the RSA algorithms that use parameters = NULL. For id-ecPublicKey the namedCurve parameter is encoded in the CBOR int. The 'subjectPublicKey' BIT STRING value field is encoded as a CBOR byte string. This specification assumes the BIT STRING has zero unused bits and the unused bits byte is omitted. Uncompressed public keys of type id-ecPublicKey are point compressed as defined in Section 2.3.3 of [\[SECG\]](#).
If a DER encoded certificate with a point compressed public key of type id-ecPublicKey is comreseed, the octets 0xfe and 0xfd are used instead of 0x02 and 0x03 in the CBOR encoding to represent a even and off y-coordinate respectively.
- o **extensions.** The 'extensions' field is encoded as a CBOR array where each extension is encoded as either a registered extension (an CBOR int followed by an optional CBOR item of any type) or a raw extension (a relative OID byte string, a bool, and a the DER encoved value of 'extnValue'). If the array contains exactly one int, the array is omitted. Extensions are encoded as specified in [Section 3.2](#). The extensions mandated to be supported by [\[RFC7925\]](#) are given special treatment.
- o **signatureValue.** The 'signatureValue' BIT STRING value field is encoded as the CBOR byte string issuerSignatureValue. This specification assumes the BIT STRING has zero unused bits and the unused bits byte is omitted. ECDSA signatures are given special treatment. For ECDSA signatures the SEQUENCE and INTEGER type and length fields are omitted and the two INTEGER value fields are padded to the fixed length $L = \text{ceil}(\log_2(n) / 8)$, where n is the size of the largest prime-order subgroup. For secp256r1, secp384r1, and secp521r1, L is 32, 48, and 66 respectively. For natively signed CBOR certificates the signatureValue is calculated over the CBOR sequence TBSCertificate.

In addition to the above fields present in X.509, the CBOR encoding introduces an additional field:

- o `cborCertificateType`. A CBOR int used to indicate the type of CBOR certificate. Currently, type can be a natively signed CBOR certificate (`cborCertificateType = 0`) or a CBOR compressed X.509 certificates (`cborCertificateType = 1`), see [Section 9.1](#).

The following Concise Data Definition Language (CDDL) defines `CBORCertificate` and `TBSCertificate`, which are encoded as CBOR Sequences [[RFC8742](#)]. The member names therefore only have documentary value.

; This defines an array, the elements of which are to be used in a CBOR Sequence:

```
CBORCertificate = [  
    TBSCertificate,  
    issuerSignatureValue : bytes,  
]
```

```
TBSCertificate = (  
    cborCertificateType : int,  
    certificateSerialNumber : bytes,  
    issuerSignatureAlgorithm : Algorithm,  
    issuer : Name,  
    validityNotBefore : bytes,  
    validityNotAfter : bytes,  
    subject : Name,  
    subjectPublicKeyAlgorithm : Algorithm,  
    subjectPublicKey : bytes,  
    extensions : [ * Extension ] / int,  
)
```

Algorithm = int / relativeOID

relativeOID = bytes

Name = [* RelativeDistinguishedName] / RelativeDistinguishedName

RelativeDistinguishedName = [+ Attribute] / text / bytes

```
Attribute = (  
    attributeType : int,  
    attributeValue : text,  
)
```

Extension = ExtensionReg // ExtensionRaw

```
ExtensionReg = (  
    extensionType : int,  
    ? extensionValue : any, ; optionality and type known from extensionType  
)
```

```
ExtensionRaw = (  
    extensionID : relativeOID,  
    ? critical : bool,  
    ? extensionValue : bytes,  
)
```


3.2. Encoding of Extensions

EDITOR'S NOTE: The current specification encodes many common extensions with a DER encoded byte string. It should be discussed if more or all commonly active extensions should be natively encoded with CBOR. Would an specific CBOR encoding have to be specified for each extension or can a general CBOR encoding that apply to all remaining extensions be specified?

This section details the encoding of the 'extensions' field. The 'extensions' field is encoded as a CBOR array where each extension is encoded as either a registered extension (an CBOR int followed by an optional CBOR item of any type) or a raw extension (a relative OID byte string, a bool, and a the DER encoved value of 'extnValue'). For registered extensions each 'extnID' field is encoded as a CBOR int (see [Section 9.3](#)), where the sign is used to encode if the extension 'critical' field. Critical extensions are encoded with a positive sign and non-critical extensions are encoded with a negative sign. If the array contains exactly one int, the array is omitted. The 'extnValue' OCTET STREAM value field is encoded as the CBOR byte string 'extensionValue' except for the extensions specified below.

The extensions mandated to be supported by [\[RFC7925\]](#) are given special treatment. Below the boolean values (cA, digitalSignature, keyAgreement, etc.) are set to 0 or 1 according to their value in the DER encoding.:

- o basicConstraints. A basic constrained with 'cA' = false is encoded as extensionType = 1, a basic constrained with 'cA' = true without 'pathLenConstraint' is encoded as extensionType = 2, and a basic constrained with 'cA' = true with 'pathLenConstraint' is encoded as extensionType = 3 followed by and int extensionValue encoding the value of 'pathLenConstraint'.
- o keyUsage. The extensionType is encoded as below. If none of the bits except digitalSignature, keyAgreement, and keyCertSign are set, the extensionValue is omitted. Otherwise the 'KeyUsage' BIT STRING is interpreted as an unsigned integer n in network byte order and encoded as a CBOR int.

$$\text{extensionType} = 4 + \text{digitalSignature} + 2 * \text{keyAgreement} + 4 * \text{keyCertSign}$$

- o extKeyUsage. extensionType is encoded as defined by [Section 9.3](#) and extensionValue is encoded as an array of ints or relativeOID where each ints or relativeOID encodes a key usage purpose (see [Section 9.4](#) for registered ints). If the array contains a single item, the array is omitted.

`extensionValue = [* int / relativeOID] / int / relativeOID`

- o `subjectAltName.extensionType` is encoded as defined by [Section 9.3](#) and `extensionValue` is encoded as an `[* (int, any)]` array where each `(int, any)` pair encodes a general name (see [Section 9.5](#)). If `subjectAltName` contains exactly one `dNSName`, the array and the `int` are omitted and `extensionValue` is the `dNSName` encoded as a CBOR text string.

Consequently:

- o A critical `basicConstraints` (`ca = 1`) without `pathLenConstraint` is encoded as the CBOR int -2.
- o A non-critical `keyUsage` with only `keyAgreement` asserted is encoded as the CBOR int 6 (`= 4 + 2`).
- o A non-critical `extKeyUsage` containing `id-kp-codeSigning` and `id-kp-OCSPSigning` is encoded as the CBOR int 12 followed by the CBOR array `[3, 6]`.
- o A non-critical `subjectAltName` containing only the `dNSName` `example.com` is encoded as the CBOR int 13 followed by the CBOR text string `"example.com"`.

Thus, the extension field of a certificate containing all of the above extensions in the given order would be encoded as the CBOR array `[-2, 6, 12, [3, 6], 13, "example.com"]`.

4. Compliance Requirements for Constrained IoT

For general purpose applications, the normative requirements of [\[RFC5280\]](#) applies. This section describes the mandatory to implement algorithms and OIDs for constrained IoT application; the values of the OIDs including certificate fields and extensions, time format, attributes in distinguished names, etc.

TODO: Write this section

5. Deployment settings

CBOR certificates can be deployed with legacy X.509 certificates and CA infrastructure. In order to verify the signature, the CBOR certificate is used to recreate the original X.509 data structure to be able to verify the signature.

For protocols like TLS/DTLS 1.2, where the handshake is sent unencrypted, the actual encoding and compression can be done at

different locations depending on the deployment setting. For example, the mapping between CBOR certificate and standard X.509 certificate can take place in a 6LoWPAN border gateway which allows the server side to stay unmodified. This case gives the advantage of the low overhead of a CBOR certificate over a constrained wireless links. The conversion to X.509 within an IoT device will incur a computational overhead, however, measured in energy this is negligible compared to the reduced communication overhead.

For the setting with constrained server and server-only authentication, the server only needs to be provisioned with the CBOR certificate and does not perform the conversion to X.509. This option is viable when client authentication can be asserted by other means.

For protocols like IKEv2, TLS/DTLS 1.3, and EDHOC, where certificates are encrypted, the proposed encoding needs to be done fully end-to-end, through adding the encoding/decoding functionality to the server.

6. Expected Certificate Sizes

The CBOR encoding of the sample certificate given in [Appendix A](#) results in the numbers shown in Figure 1. After [[RFC7925](#)] profiling, most duplicated information has been removed, and the remaining text strings are minimal in size. Therefore, the further size reduction reached with general compression mechanisms will be small, mainly corresponding to making the ASN.1 encoding more compact. The zlib number was calculated with zlib-flate.

zlib-flate -compress < cert.der > cert.compressed

+-----+-----+-----+-----+								
		RFC 7925		zlib		CBOR Certificate		
+-----+-----+-----+-----+								
	Certificate Size		314		295		138	
+-----+-----+-----+-----+								

Figure 1: Comparing Sizes of Certificates (bytes)

7. Natively Signed CBOR Certificates

The difference between CBOR compressed X.509 certificate and natively signed CBOR certificate is that the signature is calculated over the CBOR encoding of the CBOR sequence TBSCertificate rather than the DER encoded ASN.1 data. This removes entirely the need for ASN.1 DER and base64 encoding which reduces the processing in the authenticating devices and avoids known complexities with these encoding.

Natively signed CBOR certificates can be applied in devices that are only required to authenticate to natively signed CBOR certificate compatible servers. This is not a major restriction for many IoT deployments, where the parties issuing and verifying certificates can be a restricted ecosystem which not necessarily involves public CAs.

CBOR compressed X.509 certificates provides an intermediate step between [\[RFC7925\]](#) profiled X.509 certificates and natively signed CBOR certificates: An implementation of CBOR compressed X.509 certificates contains both the CBOR encoding of the X.509 certificate and the signature operations sufficient for natively signed CBOR certificates.

The natively signed approach based on DER encoded X.509 certificates described in this document has a lot of benefits. A CA can use existing ASN.1 machinery to create a DER encoded certificate, the DER encoded certificate can then be transformed to CBOR before signing.

8. Security Considerations

The CBOR profiling of X.509 certificates does not change the security assumptions needed when deploying standard X.509 certificates but decreases the number of fields transmitted, which reduces the risk for implementation errors.

Conversion between the certificate formats can be made in constant time to reduce risk of information leakage through side channels.

The mechanism in this draft does not reveal any additional information compared to X.509. Because of difference in size, it will be possible to detect that this profile is used. The gateway solution described in [Section 5](#) requires unencrypted certificates and is not recommended.

9. IANA Considerations

For all items, the 'Reference' field points to this document.

9.1. CBOR Certificate Types Registry

IANA has created a new registry titled "CBOR Certificate Types" under the new heading "CBOR Certificate". For values in the interval [-24, 23] the registration procedure is "IETF Review". For all other values the registration procedure is "Expert Review". The columns of the registry are Value, Description, and Reference, where Value is an integer, and the other columns are text strings. The initial contents of the registry are:

Value	Description
0	Natively Signed CBOR Certificate
1	CBOR Compressed X.509 Certificate

Figure 2: CBOR Certificate Types

9.2. CBOR Attribute Type Registry

IANA has created a new registry titled "CBOR Attribute Type Registry" under the new heading "CBOR Certificate". The columns of the registry are Value, X.509 Attribute Type, and Reference, where Value is an integer, and the other columns are text strings. Only positive values can be registered. For values in the interval [1, 23] the registration procedure is "IETF Review". For all other values the registration procedure is "Expert Review". The initial contents of the registry are:

Value	X.509 Attribute Type
1	id-at-commonName
2	id-at-surname
3	id-at-serialNumber
4	id-at-countryName
5	id-at-localityName
6	id-at-stateOrProvinceName
7	id-at-organizationName
8	id-at-organizationalUnitName
9	id-at-title
10	id-at-givenName
11	id-at-initials
12	id-at-generationQualifier
13	id-at-dnQualifier
14	id-at-pseudonym

Figure 3: CBOR Attribute Type Registry

9.3. CBOR Extension Type Registry

IANA has created a new registry titled "CBOR Extension Type Registry" under the new heading "CBOR Certificate". The columns of the registry are Value, X.509 Extension Type, and Reference, where Value is an integer, and the other columns are text strings. Only positive values can be registered. For values in the interval [1, 23] the

registration procedure is "IETF Review". For all other values the registration procedure is "Expert Review". The initial contents of the registry are:

Value	X.509 Extension Type	extensionValue
1	id-ce-basicConstraints (cA = false)	
2	id-ce-basicConstraints (cA = true)	
3	id-ce-basicConstraints (cA = true)	int
4	id-ce-keyUsage	int
5	id-ce-keyUsage + 1	
6	id-ce-keyUsage + 16	
7	id-ce-keyUsage + 17	
8	id-ce-keyUsage + 32	
9	id-ce-keyUsage + 33	
10	id-ce-keyUsage + 48	
11	id-ce-keyUsage + 49	
12	id-ce-extKeyUsage	[] / int / rOID
13	id-ce-subjectAltName	[] / text
14	id-ce-authorityKeyIdentifier	bytes
15	id-ce-subjectKeyIdentifier	bytes
16	id-ce-certificatePolicies	bytes
17	id-ce-cRLDistributionPoints	bytes
18	id-pe-authorityInfoAccess	bytes
19	SCT List (1.3.6.1.4.1.11129.2.4.2)	bytes
248	id-ce-nameConstraints	bytes
249	id-ce-policyConstraints	bytes
250	id-ce-inhibitAnyPolicy	bytes
251	id-ce-authorityKeyIdentifier	bytes
252	id-ce-policyMappings	bytes
253	id-ce-issuerAltName	bytes
254	id-ce-subjectDirectoryAttributes	bytes
255	id-ce-freshestCRL	bytes
256	id-pe-subjectInfoAccess	bytes

Figure 4: CBOR Extension Type Registry

9.4. CBOR Extended Key Usage Registry

IANA has created a new registry titled "CBOR Extended Key Usage Registry" under the new heading "CBOR Certificate". The columns of the registry are Value, Extended Key Usage Purpose, and Reference, where Value is an integer, and the other columns are text strings. For values in the interval [-24, 23] the registration procedure is "IETF Review". For all other values the registration procedure is "Expert Review". The initial contents of the registry are:

Value	Extended Key Usage
0	anyExtendedKeyUsage
1	id-kp-serverAuth
2	id-kp-clientAuth
3	id-kp-codeSigning
4	id-kp-emailProtection
5	id-kp-timeStamping
6	id-kp-OCSPSigning

Figure 5: CBOR Extended Key Usage Registry

9.5. CBOR Subject Alternative Name Registry

IANA has created a new registry titled "CBOR Subject Alternative Name Registry" under the new heading "CBOR Certificate". The columns of the registry are Value, Extended Key Usage Purpose, and Reference, where Value is an integer, and the other columns are text strings. For values in the interval [-24, 23] the registration procedure is "IETF Review". For all other values the registration procedure is "Expert Review". The initial contents of the registry are:

Value	Subject Alternative Name	
0	rfc822Name	text
1	dnsName	text
2	directoryName	Name
3	uniformResourceIdentifier	text
4	iPAddress	bytes

Figure 6: CBOR Subject Alternative Name Registry

9.6. CBOR Certificate Signature Algorithms Registry

IANA has created a new registry titled "CBOR Certificate Signature Algorithms" under the new heading "CBOR Certificate". For values in the interval [-24, 23] the registration procedure is "IETF Review". For all other values the registration procedure is "Expert Review". The columns of the registry are Value, X.509 Algorithm, and Reference, where Value is an integer, and the other columns are text strings. The initial contents of the registry are:

EDITOR'S NOTE: This is probably to many algorithms. All sha224, sha3, and maybe ecdsa-with-SHA1 can probably be removed.

Value	X.509 Signature Algorithm
0	sha1WithRSAEncryption
1	sha256WithRSAEncryption
2	sha384WithRSAEncryption
3	sha512WithRSAEncryption
4	id-RSASSA-PSS-SHAKE128
5	id-RSASSA-PSS-SHAKE256
6	ecdsa-with-SHA256
7	ecdsa-with-SHA384
8	ecdsa-with-SHA512
9	id-ecdsa-with-shake128
10	id-ecdsa-with-shake256
11	id-Ed25519
12	id-Ed448
13	id-alg-hss-lms-hashsig
14	id-alg-xmss
15	id-alg-xmssmt
245	sha224WithRSAEncryption
246	id-rsassa-pkcs1-v1_5-with-sha3-224
247	id-rsassa-pkcs1-v1_5-with-sha3-256
248	id-rsassa-pkcs1-v1_5-with-sha3-384
249	id-rsassa-pkcs1-v1_5-with-sha3-512
250	ecdsa-with-SHA1
251	ecdsa-with-SHA224
252	id-ecdsa-with-sha3-224
253	id-ecdsa-with-sha3-256
254	id-ecdsa-with-sha3-384
255	id-ecdsa-with-sha3-512

Figure 7: CBOR Certificate Signature Algorithms

9.7. CBOR Certificate Public Key Algorithms Registry

IANA has created a new registry titled "CBOR Certificate Public Key Algorithms" under the new heading "CBOR Certificate". For values in the interval [-24, 23] the registration procedure is "IETF Review". For all other values the registration procedure is "Expert Review". The columns of the registry are Value, X.509 Algorithm, and Reference, where Value is an integer, and the other columns are text strings. The initial contents of the registry are:

Value	X.509 Public Key Algorithm
0	rsaEncryption
1	id-ecPublicKey + secp256r1
2	id-ecPublicKey + secp384r1
3	id-ecPublicKey + secp521r1
4	id-X25519
5	id-X448
6	id-Ed25519
7	id-Ed448
8	id-alg-hss-lms-hashsig
9	id-alg-xmss
10	id-alg-xmssmt

Figure 8: CBOR Certificate Public Key Algorithms

9.8. COSE Header Parameters Registry

This document registers the following entries in the "COSE Header Parameters" registry under the "CBOR Object Signing and Encryption (COSE)" heading. The formatting and processing are the same as the corresponding x5bag, x5chain, x5t, and x5u defined in [\[I-D.ietf-cose-x509\]](#) except that the certificates are CBOR encoded instead of DER encoded.

Name	Label	Value Type	Description
c5bag	TBD1	COSE_CBOR_Cert	An ordered chain of CBOR certificates
c5chain	TBD2	COSE_CBOR_Cert	An ordered chain of CBOR certificates
c5t	TBD3	COSE_CertHash	Hash of an CBOR certificate
c5u	TBD4	uri	URI pointing to a CBOR certificate

9.9. TLS Certificate Types Registry

This document registers the following entry in the "TLS Certificate Types" registry under the "Transport Layer Security (TLS) Extensions" heading.

EDITOR'S NOTE: The TLS registrations should be discussed and approved by the TLS WG at a later stage. When COSE WG has adopted work on CBOR certificates, it could perhaps be presented in the TLS WG. The TLS WG might e.g. want a separate draft in the TLS WG.

Value	Name	Recommended	Comment
TBD3	CBOR Certificate	Y	

10. References

10.1. Normative References

- [I-D.ietf-tls-certificate-compression]
Ghedini, A. and V. Vasiliev, "TLS Certificate Compression", [draft-ietf-tls-certificate-compression-10](#) (work in progress), January 2020.
- [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-dtls13-39](#) (work in progress), November 2020.
- [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>>.
- [RFC7049] Bormann, C. and P. Hoffman, "Concise Binary Object Representation (CBOR)", [RFC 7049](#), DOI 10.17487/RFC7049, October 2013, <<https://www.rfc-editor.org/info/rfc7049>>.
- [RFC7925] Tschofenig, H., Ed. and T. Fossati, "Transport Layer Security (TLS) / Datagram Transport Layer Security (DTLS) Profiles for the Internet of Things", [RFC 7925](#), DOI 10.17487/RFC7925, July 2016, <<https://www.rfc-editor.org/info/rfc7925>>.

- [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>>.
- [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>>.
- [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>>.
- [RFC8742] Bormann, C., "Concise Binary Object Representation (CBOR) Sequences", [RFC 8742](#), DOI 10.17487/RFC8742, February 2020, <<https://www.rfc-editor.org/info/rfc8742>>.

10.2. Informative References

- [I-D.ietf-cose-x509] Schaad, J., "CBOR Object Signing and Encryption (COSE): Header parameters for carrying and referencing X.509 certificates", [draft-ietf-cose-x509-07](#) (work in progress), September 2020.
- [I-D.ietf-lake-edhoc] Selander, G., Mattsson, J., and F. Palombini, "Ephemeral Diffie-Hellman Over COSE (EDHOC)", [draft-ietf-lake-edhoc-01](#) (work in progress), August 2020.
- [RFC7228] Bormann, C., Ersue, M., and A. Keranen, "Terminology for Constrained-Node Networks", [RFC 7228](#), DOI 10.17487/RFC7228, May 2014, <<https://www.rfc-editor.org/info/rfc7228>>.
- [SECG] "Elliptic Curve Cryptography, Standards for Efficient Cryptography Group, ver. 2", 2009, <<https://secg.org/sec1-v2.pdf>>.

[X.509-IoT]

Forsby, F., Furuheid, M., Papadimitratos, P., and S. Raza,
"Lightweight X.509 Digital Certificates for the Internet
of Things.", Springer, Cham. Lecture Notes of the
Institute for Computer Sciences, Social Informatics and
Telecommunications Engineering, vol 242., July 2018,
<https://doi.org/10.1007/978-3-319-93797-7_14>.

Appendix A. Example CBOR Certificates

A.1. Example [RFC 7925](#) profiled X.509 Certificate

Example of [[RFC7925](#)] profiled X.509 certificate parsed with OpenSSL.

Certificate:

Data:

Version: 3 (0x2)
Serial Number: 128269 (0x1f50d)
Signature Algorithm: ecdsa-with-SHA256
Issuer: CN=RFC test CA
Validity
 Not Before: Jan 1 00:00:00 2020 GMT
 Not After : Feb 2 00:00:00 2021 GMT
Subject: CN=01-23-45-FF-FE-67-89-AB
Subject Public Key Info:
 Public Key Algorithm: id-ecPublicKey
 Public-Key: (256 bit)
 pub:
 04:ae:4c:db:01:f6:14:de:fc:71:21:28:5f:dc:7f:
 5c:6d:1d:42:c9:56:47:f0:61:ba:00:80:df:67:88:
 67:84:5e:e9:a6:9f:d4:89:31:49:da:e3:d3:b1:54:
 16:d7:53:2c:38:71:52:b8:0b:0d:f3:e1:af:40:8a:
 95:d3:07:1e:58
 ASN1 OID: prime256v1
 NIST CURVE: P-256
X509v3 extensions:
 X509v3 Key Usage:
 Digital Signature
Signature Algorithm: ecdsa-with-SHA256
 30:44:02:20:37:38:73:ef:87:81:b8:82:97:ef:23:5c:1f:ac:
 cf:62:da:4e:44:74:0d:c2:a2:e6:a3:c6:c8:82:a3:23:8d:9c:
 02:20:3a:d9:35:3b:a7:88:68:3b:06:bb:48:fe:ca:16:ea:71:
 17:17:34:c6:75:c5:33:2b:2a:f1:cb:73:38:10:a1:fc

The DER encoding of the above certificate is 314 bytes.


```

30 82 01 36 30 81 DE A0 03 02 01 02 02 03 01 F5 0D 30 0A 06 08 2A 86 48
CE 3D 04 03 02 30 16 31 14 30 12 06 03 55 04 03 0C 0B 52 46 43 20 74 65
73 74 20 43 41 30 1E 17 0D 32 30 30 31 30 31 30 30 30 30 30 5A 17 0D
32 31 30 32 30 32 30 30 30 30 30 30 5A 30 22 31 20 30 1E 06 03 55 04 03
0C 17 30 31 2D 32 33 2D 34 35 2D 46 46 2D 46 45 2D 36 37 2D 38 39 2D 41
42 30 59 30 13 06 07 2A 86 48 CE 3D 02 01 06 08 2A 86 48 CE 3D 03 01 07
03 42 00 04 AE 4C DB 01 F6 14 DE FC 71 21 28 5F DC 7F 5C 6D 1D 42 C9 56
47 F0 61 BA 00 80 DF 67 88 67 84 5E E9 A6 9F D4 89 31 49 DA E3 D3 B1 54
16 D7 53 2C 38 71 52 B8 0B 0D F3 E1 AF 40 8A 95 D3 07 1E 58 A3 0F 30 0D
30 0B 06 03 55 1D 0F 04 04 03 02 07 80 30 0A 06 08 2A 86 48 CE 3D 04 03
02 03 47 00 30 44 02 20 37 38 73 EF 87 81 B8 82 97 EF 23 5C 1F AC CF 62
DA 4E 44 74 0D C2 A2 E6 A3 C6 C8 82 A3 23 8D 9C 02 20 3A D9 35 3B A7 88
68 3B 06 BB 48 FE CA 16 EA 71 17 17 34 C6 75 C5 33 2B 2A F1 CB 73 38 10
A1 FC

```

A.1.1. Example CBOR Certificate Compression

The CBOR certificate compression of the X.509 in CBOR diagnostic format is:

/This defines a CBOR Sequence ([RFC 8742](#)):/

```

1,
h'01f50d',
6,
"RFC test CA",
h'2B044180',
h'2D543300',
h'0123456789AB',
1,
h'02ae4cdb01f614defc7121285fdc7f5c6d1d42c95647f061ba
0080df678867845e',
5,
h'373873EF8781B88297EF235C1FACCF62DA4E44740DC2A2E6A3
C6C882A3238D9C3AD9353BA788683B06BB48FECA16EA711717
34C675C5332B2AF1CB733810A1FC'

```

The CBOR encoding (CBOR sequence) of the CBOR certificate is 138 bytes.

```

01 43 01 F5 0D 2A 6B 52 46 43 20 74 65 73 74 20 43 41 44 2B 04 41 80 44
2D 54 33 00 46 01 23 45 67 89 AB 36 58 21 02 AE 4C DB 01 F6 14 DE FC 71
21 28 5F DC 7F 5C 6D 1D 42 C9 56 47 F0 61 BA 00 80 DF 67 88 67 84 5E 05
58 40 37 38 73 EF 87 81 B8 82 97 EF 23 5C 1F AC CF 62 DA 4E 44 74 0D C2
A2 E6 A3 C6 C8 82 A3 23 8D 9C 3A D9 35 3B A7 88 68 3B 06 BB 48 FE CA 16
EA 71 17 17 34 C6 75 C5 33 2B 2A F1 CB 73 38 10 A1 FC

```


A.1.2. Example: Natively Signed CBOR Certificate

The corresponding natively signed CBOR certificate in CBOR diagnostic format is identical except for type and signatureValue.

/This defines a CBOR Sequence ([RFC 8742](#)):/

```
0,
h'01f50d',
6,
"RFC test CA",
h'2B044180',
h'2D543300',
h'0123456789AB',
1,
h'02ae4cdb01f614defc7121285fdc7f5c6d1d42c95647f061
ba0080df678867845e',
5,
h'7F10A063DA8DB2FD49414440CDF85070AC22A266C7F1DFB1
577D9A35A295A8742E794258B76968C097F85542322A0796
0199C13CC0220A9BC729EF2ECA638CFE'
```

The CBOR encoding (CBOR sequence) of the CBOR certificate is 138 bytes.

```
00 43 01 F5 0D 2A 6B 52 46 43 20 74 65 73 74 20 43 41 44 2B 04 41 80 44
2D 54 33 00 46 01 23 45 67 89 AB 36 58 21 02 AE 4C DB 01 F6 14 DE FC 71
21 28 5F DC 7F 5C 6D 1D 42 C9 56 47 F0 61 BA 00 80 DF 67 88 67 84 5E 05
58 40 7F 10 A0 63 DA 8D B2 FD 49 41 44 40 CD F8 50 70 AC 22 A2 66 C7 F1
DF B1 57 7D 9A 35 A2 95 A8 74 2E 79 42 58 B7 69 68 C0 97 F8 55 42 32 2A
07 96 01 99 C1 3C C0 22 0A 9B C7 29 EF 2E CA 63 8C FE
```

A.2. Example HTTPS X.509 Certificate

The DER encoding of the tools.ietf.org certificate is 1647 bytes.

```
30 82 06 6b 30 82 05 53 a0 03 02 01 02 02 09 00 a6 a5 5c 87 0e 39 b4 0e
30 0d 06 09 2a 86 48 86 f7 0d 01 01 0b 05 00 30 81 c6 31 0b 30 09 06 03
55 04 06 13 02 55 53 31 10 30 0e 06 03 55 04 08 13 07 41 72 69 7a 6f 6e
61 31 13 30 11 06 03 55 04 07 13 0a 53 63 6f 74 74 73 64 61 6c 65 31 25
30 23 06 03 55 04 0a 13 1c 53 74 61 72 66 69 65 6c 64 20 54 65 63 68 6e
6f 6c 6f 67 69 65 73 2c 20 49 6e 63 2e 31 33 30 31 06 03 55 04 0b 13 2a
68 74 74 70 3a 2f 2f 63 65 72 74 73 2e 73 74 61 72 66 69 65 6c 64 74 65
63 68 2e 63 6f 6d 2f 72 65 70 6f 73 69 74 6f 72 79 2f 31 34 30 32 06 03
55 04 03 13 2b 53 74 61 72 66 69 65 6c 64 20 53 65 63 75 72 65 20 43 65
72 74 69 66 69 63 61 74 65 20 41 75 74 68 6f 72 69 74 79 20 2d 20 47 32
30 1e 17 0d 32 30 31 30 30 31 31 39 33 38 33 36 5a 17 0d 32 31 31 31 30
```


Internet-DraftCBOR Encoding of X.509 Certificates (CBOR CerNovember 2020

32 31 39 33 38 33 36 5a 30 3e 31 21 30 1f 06 03 55 04 0b 13 18 44 6f 6d
61 69 6e 20 43 6f 6e 74 72 6f 6c 20 56 61 6c 69 64 61 74 65 64 31 19 30
17 06 03 55 04 03 0c 10 2a 2e 74 6f 6f 6c 73 2e 69 65 74 66 2e 6f 72 67
30 82 01 22 30 0d 06 09 2a 86 48 86 f7 0d 01 01 01 05 00 03 82 01 0f 00
30 82 01 0a 02 82 01 01 00 b1 e1 37 e8 eb 82 d6 89 fa db f5 c2 4b 77 f0
2c 4a de 72 6e 3e 13 60 d1 a8 66 1e c4 ad 3d 32 60 e5 f0 99 b5 f4 7a 7a
48 55 21 ee 0e 39 12 f9 ce 0d ca f5 69 61 c7 04 ed 6e 0f 1d 3b 1e 50 88
79 3a 0e 31 41 16 f1 b1 02 64 68 a5 cd f5 4a 0a ca 99 96 35 08 c3 7e 27
5d d0 a9 cf f3 e7 28 af 37 d8 b6 7b dd f3 7e ae 6e 97 7f f7 ca 69 4e cc
d0 06 df 5d 27 9b 3b 12 e7 e6 fe 08 6b 52 7b 82 11 7c 72 b3 46 eb c1 e8
78 b8 0f cb e1 eb bd 06 44 58 dc 83 50 b2 a0 62 5b dc 81 b8 36 e3 9e 7c
79 b2 a9 53 8a e0 0b c9 4a 2a 13 39 31 13 bd 2c cf a8 70 cf 8c 8d 3d 01
a3 88 ae 12 00 36 1d 1e 24 2b dd 79 d8 53 01 26 ed 28 4f c9 86 94 83 4e
c8 e1 14 2e 85 b3 af d4 6e dd 69 46 af 41 25 0e 7a ad 8b f2 92 ca 79 d9
7b 32 4f f7 77 e8 f9 b4 4f 23 5c d4 5c 03 ae d8 ab 3a ca 13 5f 5d 5d 5d
a1 02 03 01 00 01 a3 82 02 e1 30 82 02 dd 30 0c 06 03 55 1d 13 01 01 ff
04 02 30 00 30 1d 06 03 55 1d 25 04 16 30 14 06 08 2b 06 01 05 05 07 03
01 06 08 2b 06 01 05 05 07 03 02 30 0e 06 03 55 1d 0f 01 01 ff 04 04 03
02 05 a0 30 3d 06 03 55 1d 1f 04 36 30 34 30 32 a0 30 a0 2e 86 2c 68 74
74 70 3a 2f 2f 63 72 6c 2e 73 74 61 72 66 69 65 6c 64 74 65 63 68 2e 63
6f 6d 2f 73 66 69 67 32 73 31 2d 32 34 32 2e 63 72 6c 30 63 06 03 55 1d
20 04 5c 30 5a 30 4e 06 0b 60 86 48 01 86 fd 6e 01 07 17 01 30 3f 30 3d
06 08 2b 06 01 05 05 07 02 01 16 31 68 74 74 70 3a 2f 2f 63 65 72 74 69
66 69 63 61 74 65 73 2e 73 74 61 72 66 69 65 6c 64 74 65 63 68 2e 63 6f
6d 2f 72 65 70 6f 73 69 74 6f 72 79 2f 30 08 06 06 67 81 0c 01 02 01 30
81 82 06 08 2b 06 01 05 05 07 01 01 04 76 30 74 30 2a 06 08 2b 06 01 05
05 07 30 01 86 1e 68 74 74 70 3a 2f 2f 6f 63 73 70 2e 73 74 61 72 66 69
65 6c 64 74 65 63 68 2e 63 6f 6d 2f 30 46 06 08 2b 06 01 05 05 07 30 02
86 3a 68 74 74 70 3a 2f 2f 63 65 72 74 69 66 69 63 61 74 65 73 2e 73 74
61 72 66 69 65 6c 64 74 65 63 68 2e 63 6f 6d 2f 72 65 70 6f 73 69 74 6f
72 79 2f 73 66 69 67 32 2e 63 72 74 30 1f 06 03 55 1d 23 04 18 30 16 80
14 25 45 81 68 50 26 38 3d 3b 2d 2c be cd 6a d9 b6 3d b3 66 63 30 2b 06
03 55 1d 11 04 24 30 22 82 10 2a 2e 74 6f 6f 6c 73 2e 69 65 74 66 2e 6f
72 67 82 0e 74 6f 6f 6c 73 2e 69 65 74 66 2e 6f 72 67 30 1d 06 03 55 1d
0e 04 16 04 14 ad 8a b4 1c 07 51 d7 92 89 07 b0 b7 84 62 2f 36 55 7a 5f
4d 30 82 01 06 06 0a 2b 06 01 04 01 d6 79 02 04 02 04 81 f7 04 81 f4 00
f2 00 77 00 f6 5c 94 2f d1 77 30 22 14 54 18 08 30 94 56 8e e3 4d 13 19
33 bf df 0c 2f 20 0b cc 4e f1 64 e3 00 00 01 74 e5 ac 71 13 00 00 04 03
00 48 30 46 02 21 00 8c f5 48 52 ce 56 35 43 39 11 cf 10 cd b9 1f 52 b3
36 39 22 3a d1 38 a4 1d ec a6 fe de 1f e9 0f 02 21 00 bc a2 25 43 66 c1
9a 26 91 c4 7a 00 b5 b6 53 ab bd 44 c2 f8 ba ae f4 d2 da f2 52 7c e6 45
49 95 00 77 00 5c dc 43 92 fe e6 ab 45 44 b1 5e 9a d4 56 e6 10 37 fb d5
fa 47 dc a1 73 94 b2 5e e6 f6 c7 0e ca 00 00 01 74 e5 ac 72 3c 00 00 04
03 00 48 30 46 02 21 00 a5 e0 90 6e 63 e9 1d 4f dd ef ff 03 52 b9 1e 50
89 60 07 56 4b 44 8a 38 28 f5 96 dc 6b 28 72 6d 02 21 00 fc 91 ea ed 02
16 88 66 05 4e e1 8a 2e 53 46 c4 cc 51 fe b3 fa 10 a9 1d 2e db f9 91 25
f8 6c e6 30 0d 06 09 2a 86 48 86 f7 0d 01 01 0b 05 00 03 82 01 01 00 14
04 3f a0 be d2 ee 3f a8 6e 3a 1f 78 8e a0 4c 35 53 0f 11 06 1f ff 60 a1


```
6d 0b 83 e9 d9 2a db b3 3f 9d b3 d7 e0 59 4c 19 a8 e4 19 a5 0c a7 70 72
77 63 d5 fe 64 51 0a d2 7a d6 50 a5 8a 92 38 ec cb 2f 0f 5a c0 64 58 4d
5c 06 b9 73 63 68 27 8b 89 34 dc 79 c7 1d 3a fd 34 5f 83 14 41 58 49 80
68 29 80 39 8a 86 72 69 cc 79 37 ce e3 97 f7 dc f3 95 88 ed 81 03 29 00
d2 a2 c7 ba ab d6 3a 8e ca 09 0b d9 fb 39 26 4b ff 03 d8 8e 2d 3f 6b 21
ca 8a 7d d8 5f fb 94 ba 83 de 9c fc 15 8d 61 fa 67 2d b0 c7 db 3d 25 0a
41 4a 85 d3 7f 49 46 37 3c f4 b1 75 d0 52 f3 dd c7 66 f1 4b fd aa 00 ed
bf e4 7e ed 01 ec 7b e4 f6 46 fc 31 fd 72 fe 03 d2 f2 65 af 4d 7e e2 81
9b 7a fd 30 3c f5 52 f4 05 34 a0 8a 3e 19 41 58 c8 a8 e0 51 71 84 09 15
ae ec a5 77 75 fa 18 f7 d5 77 d5 31 cc c7 2d
```

[A.2.1.](#) Example CBOR Certificate Compression

The CBOR certificate compression of the X.509 in CBOR diagnostic format is:

/This defines a CBOR Sequence ([RFC 8742](#)):/

```

1,
h'A6A55C870E39B40E',
0,
[
  [4, "US"],
  [6, "Arizona"],
  [5, "Scottsdale"],
  [7, "Starfield Technologies, Inc."],
  [8, "http://certs.starfieldtech.com/repository/"],
  [1, "Starfield Secure Certificate Authority - G2"]
],
h'2D3EE7F6',
h'2F98B716',
[
  [8, "Domain Control Validated"],
  [-1, "*.tools.ietf.org"]
],
0,
h'3082010A0282010100B1E137E8EB82D689FADBF5C24B77F02C4ADE726E3E1360D1A8661EC4AD3D3260E5F099B5F47A
[
  -1,
  12, [ 1, 2 ],
  -4, 5,
  17,
h'30343032a030a02e862c687474703a2f2f63726c2e737461726669656c64746563682e636f6d2f736669673273312
  16,
h'305A304E060B086480186FD6E01071701303F303D06082B060105050702011631687474703A2F2F63657274696669
  18,
h'3074302A06082B06010505073001861E687474703A2F2F6F6373702E737461726669656C64746563682E636F6D2F30
  14, h'30168014254581685026383D3B2D2CBECD6AD9B63DB36663',
  13, [ 1, "*.tools.ietf.org", 1, "tools.ietf.org" ],
  15, h'0414AD8AB41C0751D7928907B0B784622F36557A5F4D',
  19,
h'0481F400F2007700F65C942FD1773022145418083094568EE34D131933BFDF0C2F200BCC4EF164E300000174E5AC7
],
h'14043FA0BED2EE3FA86E3A1F788EA04C35530F11061FFF60A16D0B83E9D92ADBB33F9DB3D7E0594C19A8E419A50CA7

```

The CBOR encoding (CBOR sequence) of the CBOR certificate is 1374 bytes.

[Appendix B](#). X.509 Certificate Profile, ASN.1

EDITOR'S NOTE: The ASN.1 below is not up to date with the rest of the

specification. The below ASN.1 for [RFC 7925](#) profile should be in [draft-ietf-uta-tls13-iot-profile](#) instead. If CBOR Certificates support a large subset of [RFC 5280](#), we should probably not duplicate all the ASN.1 in that document. Should be discussed what kind and how much (if any) ASN.1 this document needs. If possible, one option would be to have ASN.1 for the restrictions compared to [RFC 5280](#).

IOTCertificate DEFINITIONS EXPLICIT TAGS ::= BEGIN

```
Certificate ::= SEQUENCE {
    tbsCertificate      TBSCertificate,
    signatureAlgorithm  AlgorithmIdentifier,
    signatureValue      BIT STRING
}
```

```
TBSCertificate ::= SEQUENCE {
    version             [0] INTEGER {v3(2)},
    serialNumber         INTEGER (1..MAX),
    signature            AlgorithmIdentifier,
    issuer               Name,
    validity             Validity,
    subject              Name,
    subjectPublicKeyInfo SubjectPublicKeyInfo,
    extensions           [3] Extensions OPTIONAL
}
```

Name ::= SEQUENCE SIZE (1) OF DistinguishedName

DistinguishedName ::= SET SIZE (1) OF CommonName

```
CommonName ::= SEQUENCE {
    type          OBJECT IDENTIFIER (id-at-commonName),
    value         UTF8String
}
```

```
Validity ::= SEQUENCE {
    notBefore      UTCTime,
    notAfter       UTCTime
}
```

```
SubjectPublicKeyInfo ::= SEQUENCE {
    algorithm          AlgorithmIdentifier,
    subjectPublicKey    BIT STRING
}
```

```
AlgorithmIdentifier ::= SEQUENCE {
    algorithm          OBJECT IDENTIFIER,
    parameters        ANY DEFINED BY algorithm OPTIONAL }
}
```

Extensions ::= SEQUENCE SIZE (1..MAX) OF Extension

```
Extension ::= SEQUENCE {
    extnId            OBJECT IDENTIFIER,
    critical           BOOLEAN DEFAULT FALSE,
```



```
    extnValue      OCTET STRING
  }

  id-at-commonName  OBJECT IDENTIFIER ::=
    {joint-iso-itu-t(2) ds(5) attributeType(4) 3}

END
```

Acknowledgments

The authors want to thank Henk Birkholz, Carsten Bormann, Russ Housley, Ilari Liusvaara, Laurence Lundblade, Thomas Peterson, Michael Richardson, Jim Schaad, and Rene Struik for reviewing and commenting on intermediate versions of the draft.

Authors' Addresses

Shahid Raza
RISE AB

Email: shahid.raza@ri.se

Joel Hoeglund
RISE AB

Email: joel.hoglund@ri.se

Goeran Selander
Ericsson AB

Email: goran.selander@ericsson.com

John Preuss Mattsson
Ericsson AB

Email: john.mattsson@ericsson.com

Martin Furuhed
Nexus Group

Email: martin.furuhed@nexusgroup.com

