C509 Test Vectors

draft-lspm-cose-c509-test-vectors-00

Lijun Liao, NIO
Göran Selander, John Preuß Mattsson, Ericsson AB

IETF 124, COSE WG, Montreal, Nov 04, 2024

Summary

- CBOR encoding of X.509 (C509) certificates in AD review
  - draft-ietf-cose-cbor-encoded-certs-15

- Request for C509 test vectors to support implementation and interop. testing
  - this draft

- Includes X.509 certificates and corresponding C509 certificates with different types of:
  - subject public keys
  - signature algorithms
  - subject attributes
  - certificate extensions
- C509 certificate (signing) requests
- C509 certificate request templates

Disclaimer: No PQ algorithms yet.
May be this or separate draft.

Recap C509 Certificates 1(2)

```
C509Certificate = [
   TBSCertificate,
   issuerSignatureValue : any,
]

TBSCertificate = (
   c509CertificateType: int,
   certificateSerialNumber: ~biguint,
   issuerSignatureAlgorithm: AlgorithmIdentifier,
   issuer: Name / null,
   validityNotBefore: ~time,
   validityNotAfter: ~time / null,
   subject: Name,
   subjectPublicKeyAlgorithm: AlgorithmIdentifier,
   subjectPublicKey: any .ne undefined,
   extensions: Extensions,
)
```

See Section 3.1 of draft-ietf-cose-cbor-encoded-cert-15

Recap C509 Certificates 2(2)

```
Name = [ * Attribute ] / SpecialText

Attribute = (( attributeType: int, attributeValue: SpecialText ) //
             ( attributeType: ~oid, attributeValue: bytes ))

SpecialText = text / bytes / tag

AlgorithmIdentifier = int / ~oid /
                      [ algorithm: ~oid, parameters: bytes ]

Extensions = [ * Extension ] / int

Extension = (( extensionID: int, extensionValue: any .ne undefined ) //
             ( extensionID: ~oid, ? critical: true,
              extensionValue: bytes ))
```

See Section 3.1 of draft-ietf-cose-cbor-encoded-cert-15

Example: C509 Certificate, Type 3

```
C509Certificate = [ 3,                      / Signed ASN.1 DER /
                    h'01f50d',              / serialNumber /
                    0,                      / signatureAlgorithm (ecdsa-with-SHA256) /
                    "RFC test CA",          / issuer /
                    1672531200,             / notBefore (2023-01-01T00:00:00Z) /
                    1767225600,             / notAfter (2026-01-01T00:00:00Z) /
                    48(h'0123456789AB'),    / subject, EUI-64 /
                    1,                      / subjectPublicKeyAlgorithm (P256)/
                    h'FEB1216AB96E5B3B3340F5BDF02E693F16213A04525ED44450
                      B1019C2DFD3838AB',
                    1,                      / single extension:
                                              non-critical keyUsage
                                              digitalSignature /
                    h'D4320B1D6849E309219D30037E138166F2508247DDDAE76CCE
                      EA55053C108E90D551F6D60106F1ABB484CFBE6256C178E4AC
                      3314EA19191E8B607DA5AE3BDA16' ]
```

CBOR encoding of RFC 7925 compliant X.509 certificate.

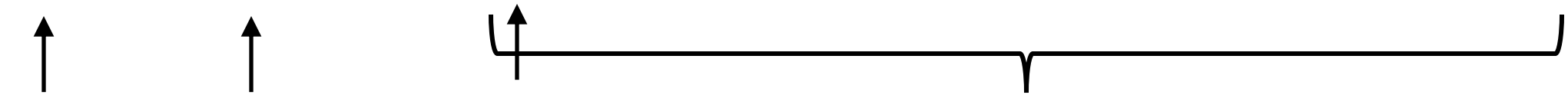
Example: C509 Certificate, Type 2

```
C509Certificate = [2,                       / Signed CBOR /
                   h'01f50d',               / serialNumber /
                   0,                       / signatureAlgorithm (ecdsa-with-SHA256) /
                   "RFC test CA",           / issuer /
                   1672531200,              / notBefore (2023-01-01T00:00:00Z) /
                   1767225600,              / notAfter (2026-01-01T00:00:00Z) /
                   48(h'0123456789AB'),     / subject, EUI-64 /
                   1,                       / subjectPublicKeyAlgorithm (P256)/
                   h'FEB1216AB96E5B3B3340F5BDF02E693F16213A04525ED44450
                     B1019C2DFD3838AB',
                   1,                       / single extension:
                                              non-critical keyUsage
                                              digitalSignature /
                   h'EB0D472731F689BC00F5880B12C68B3F9FD38B23FADFCA2095
                     0F3F241B60A202579CAC28CD3B7494D5FA5D8BBAB4600357E5
                     50AB9FA9A65D9BA2B3B82E668CC6' ]
```

Identical to previous slide, except first field (type) and last field (signature).

Test Vector 2.4 (excl. extensions and signature)

```
 0: 02                # [0]. certificate type=2
 1: 41                # [1]. serial number=byte[1]
 2:   01
 3: 0C                # [2]. signature alg=12: Ed25519
 4: F6                # [3]. issuer=<null>
 5: 1A 67748580       # [4]. notBefore=1735689600: 2025-01-01T00:00:00Z
10: 1A 6B36EC7F       # [5]. notAfter=1798761599: 2026-12-31T23:59:59Z
15: 67                # [6]. subject=char[7]
16:   74657374206361   # "test ca"
23: 0A                # [7]. subjectPublicKeyAlg=10: Ed25519
24: 58 20             # [8]. subject public key=byte[32]
26:   5A9414AC56D1B6AF0C966FC53B9476B5C95D0EEAAEF764D9EFE86DB7320C
56:   36E1
```



byte offset

CBOR plain hex

index in CBOR sequence

annotated hex

continued on next slide

Test Vector 2.4 (extensions and signature)

```
 58: 88                # [9]. extensions=array[8]
                         # extension[0]
 59:     01                # type=1: SubjectKeyIdentifier
 60:     54                # value=byte[20]
 61:       C16DE726347331107520B4ED9ED0088023A56033
                         # extension[1]
 81:     21                # type=-2: KeyUsage, critical
 82:     18 60             # value=96: [keyCertSign, cRLSign]
                         # extension[2]
 84:     03                # type=3: SubjectAlternativeName
 85:     67                # DNS, value=char[7]
 86:       6162632E636F6D   # "abc.com"
                         # extension[3]
 93:     23                # type=-4: BasicConstraints, critical
 94:     20                # value=-1: CA: true, pathLenConstraint:
                           # unlimited
 95: 58 40             # [10]. signature value=byte[64]
 97:   AA0CF7A8B267EDE76B0C2CC0240A60587700D749A959EE3D96B59E13DEF9
127:   F137DC51FB9E460A5EBF7F0C659365EBE32B44CCA4A2AC67E3D18A867C36
157:   400E2C01
```

Test Vector 3.3 (excl. extensions and signature)

```
 0: 03                # [0]. certificate type=3
 1: 42                # [1]. serial number=byte[2]
 2:   1234
 4: 00                # [2]. signature alg=0: ecdsa-with-sha256
 5: F6                # [3]. issuer=<null>
 6: 1A 6775D700       # [4]. notBefore=1735776000: 2025-01-02T00:00:00Z
11: 1A 69570A80       # [5]. notAfter=1767312000: 2026-01-02T00:00:00Z
16: 48                # [6]. subject=byte[8]
17:   1234567890ABCDEF
25: 01                # [7]. subjectPublicKeyAlg=1: EC public key with
                      #      curve secp256r1
26: 58 21             # [8]. subject public key=byte[33]
28:   FDF413596A87125995B4E0D8B7BEFBC4D6EDB11F61AF08AB32408D4FF9F9
58:   078DDB
```

continued on next slide

Test Vector 3.3 (extensions and signature)

```
 61: 88                 # [9]. extensions=array[8]
                          # extension[0]
 62:    01                  # type=1: SubjectKeyIdentifier
 63:    54                  # value=byte[20]
 64:      07E12C4CACE95C2285EC4A5B05A4A2BB0EC87A7A
                          # extension[1]
 84:    02                  # type=2: KeyUsage
 85:    18 60               # value=96: [keyCertSign, cRLSign]
                          # extension[2]
 87:    23                  # type=-4: BasicConstraints, critical
 88:    20                  # value=-1: CA: true, pathLenConstraint:
                            # unlimited
                          # extension[3]
 89:    08                  # type=8: ExtendedKeyUsage
 90:    01                  # 1: serverAuth
 91: 58 40              # [10]. signature value=byte[64]
 93:   8A25E8AABBA4B19B8E0D1596A476C2C42F5068F5F3457606806E2F284A22
123:   E6E7A799B809F4364246E7A093B3CC10CE28B5ED9AC5FEE29542483B85E6
153:   243BC13F
```

C509 Certificate Request

```
C509CertificateRequest = [
   TBSCertificateRequest,
   subjectSignatureValue: any,
]

TBSCertificateRequest = (
   c509CertificateRequestType: int,
   subjectSignatureAlgorithm: AlgorithmIdentifier,
   subject: Name,
   subjectPublicKeyAlgorithm: AlgorithmIdentifier,
   subjectPublicKey: any .ne undefined,
   extensionsRequest: Extensions,
)
```

See Section 4 of draft-ietf-cose-cbor-encoded-cert-15

Test Vector 8.1

```
  0: 00              # [0]. c509CertificateRequestType=0
  1: 00              # [1]. subjectSignatureAlgorithm=0:
                     #      ecdsa-with-sha256
  2: 65              # [2]. subject=char[5]
  3:   6563647361     # "ecdsa"
  8: 01              # [3]. subjectPublicKeyAlg=1: EC public key with
                     #      curve secp256r1
  9: 58 41           # [4]. subject public key=byte[65]
 11:   04F413596A87125995B4E0D8B7BEFBC4D6EDB11F61AF08AB32408D4FF9F9
 41:   078DDBAB3635AFD496D5656A22EFDC3D59C4482A99836BB358FBF4CA78D3
 71:   930436C857
 76: 82              # [5]. extensions=array[2]
                       # extension[0]
 77:     03            # type=3: SubjectAlternativeName
 78:     67            # DNS, value=char[7]
 79:       6162632E636F6D  # "abc.com"
 86: 58 40           # [6]. signature value=byte[64]
 88:   8A25E8AABBA4B19B8E0D1596A476C2C42F5068F5F3457606806E2F284A22
118:   E6E7C2AEF4221B91E3D99BCE70E3755804EAB51D78CD76355F13448D9D5F
148:   C4EC35BC
```

C509 Certificate Request Template

```
C509CertificateRequestTemplate = [
   c509CertificateRequestTemplateType: int,
   c509CertificateRequestType: [+ int] / undefined,
   subjectSignatureAlgorithm: [+ AlgorithmIdentifier] / undefined,
   subject: NameTemplate,
   subjectPublicKeyAlgorithm: [+ AlgorithmIdentifier] / undefined,
   subjectPublicKey: undefined
   extensionsRequest: ExtensionsTemplate,
]
```

See Section 4 of draft-ietf-cose-cbor-encoded-cert-15

Test Vector 10.2

```
 0: 00                # [0]. c509CertificateRequestTemplateType=0
 1: 81                # [1]. c509CertificateRequestType=array[1]
 2:   00                # 0
 3: 81                # [2]. subjectSignatureAlgorithm=array[1]
 4:   00                # [0]=0: ecdsa-with-sha256
 5: 82                # [3]. subject=array[2], 1 attribute
                        # attribute[0]
 6:     01                # type=1: commonName
 7:     F7                # value=<undefined>
 8: 81                # [4]. subjectPublicKeyAlgorithm=array[1]
 9:   01                # [0]=1: EC public key with curve secp256r1
10: F7                # [5]. subjectPublicKey=<undefined>
11: 82                # [6]. extensions=array[2]
                        # extension[0]
12:     03                # type=3: SubjectAlternativeName
13:     F7                # value=<undefined>
```

Thanks!

Questions / comments?