ACE Working Group
Internet-Draft
Intended status: Standards Track
Expires: December 31, 2017

M. Jones
Microsoft
E. Wahlstroem

S. Erdtman
Spotify AB
H. Tschofenig
ARM Ltd.

June 29, 2017

CBOR Web Token (CWT)
draft-ietf-ace-cbor-web-token-06

Abstract

CBOR Web Token (CWT) is a compact means of representing claims to be transferred between two parties. The claims in a CWT are encoded in the Concise Binary Object Representation (CBOR) and CBOR Object Signing and Encryption (COSE) is used for added application layer security protection. A claim is a piece of information asserted about a subject and is represented as a name/value pair consisting of a claim name and a claim value. CWT is derived from JSON Web Token (JWT), but uses CBOR rather than JSON.

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 http://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 31, 2017.

Table of Contents

1. Introduction

The JSON Web Token (JWT) [RFC7519] is a standardized security token format that has found use in OAuth 2.0 and OpenID Connect deployments, among other applications. JWT uses JSON Web Signature (JWS) [RFC7515] and JSON Web Encryption (JWE) [RFC7516] to secure the contents of the JWT, which is a set of claims represented in JSON. The use of JSON for encoding information is popular for Web and native applications, but it is considered inefficient for some Internet of Things (IoT) systems that use low power radio technologies.

An alternative encoding of claims is defined in this document. Instead of using JSON, as provided by JWTs, this specification uses CBOR [RFC7049] and calls this new structure "CBOR Web Token (CWT)", which is a compact means of representing secured claims to be transferred between two parties. CWT is closely related to JWT. It references the JWT claims and both its name and pronunciation are derived from JWT. To protect the claims contained in CWTs, the CBOR Object Signing and Encryption (COSE) [I-D.ietf-cose-msg] specification is used.

The suggested pronunciation of CWT is the same as the English word "cot".

1.1. CBOR Related Terminology

In JSON, maps are called objects and only have one kind of map key: a string. CBOR uses strings, negative integers, and unsigned integers as map keys. The integers are used for compactness of encoding and easy comparison. The inclusion of strings allows for an additional range of short encoded values to be used.

2. 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 "Key words for use in RFCs to Indicate Requirement Levels" [RFC2119].

This document reuses terminology from JWT [RFC7519] and COSE [I-D.ietf-cose-msg].

StringOrURI
: The "StringOrURI" term has the same meaning, syntax, and processing rules as the "StringOrUri" term defined in Section 2 of JWT [RFC7519], except that it uses a CBOR text string instead of a JSON string value.

NumericDate
: The "NumericDate" term has the same meaning, syntax, and processing rules as the "NumericDate" term defined in Section 2 of JWT [RFC7519], except that the CBOR numeric date representation (from Section 2.4.1 of [RFC7049]) is used. The encoding is modified so that the leading tag 1 (epoch-based date/time) MUST be omitted.

Claim Name
: The human-readable name used to identify a claim.

Claim Key
: The CBOR map key used to identify a claim.

Claim Value
: The CBOR map value representing the value of the claim.

CWT Claims Set
: The CBOR map that contains the claims conveyed by the CWT.

3. Claims

The set of claims that a CWT must contain to be considered valid is context dependent and is outside the scope of this specification. Specific applications of CWTs will require implementations to understand and process some claims in particular ways. However, in the absence of such requirements, all claims that are not understood by implementations MUST be ignored.

To keep CWTs as small as possible, the Claim Keys are represented using integers or text strings. Section 4 summarizes all keys used to identify the claims defined in this document.

3.1. Registered Claims

None of the claims defined below are intended to be mandatory to use or implement. They rather provide a starting point for a set of useful, interoperable claims. Applications using CWTs should define which specific claims they use and when they are required or optional.

3.1.1. iss (Issuer) Claim

The "iss" (issuer) claim has the same meaning, syntax, and processing rules as the "iss" claim defined in Section 4.1.1 of JWT [RFC7519], except that the value is of type StringOrURI. The Claim Key 1 is used to identify this claim.

3.1.2. sub (Subject) Claim

The "sub" (subject) claim has the same meaning, syntax, and processing rules as the "sub" claim defined in Section 4.1.2 of JWT [RFC7519], except that the value is of type StringOrURI. The Claim Key 2 is used to identify this claim.

3.1.3. aud (Audience) Claim

The "aud" (audience) claim has the same meaning, syntax, and processing rules as the "aud" claim defined in Section 4.1.3 of JWT [RFC7519], except that the value is of type StringOrURI. The Claim Key 3 is used to identify this claim.

3.1.4. exp (Expiration Time) Claim

The "exp" (expiration time) claim has the same meaning, syntax, and processing rules as the "exp" claim defined in Section 4.1.4 of JWT [RFC7519], except that the value is of type NumericDate. The Claim Key 4 is used to identify this claim.

3.1.5. nbf (Not Before) Claim

The "nbf" (not before) claim has the same meaning, syntax, and processing rules as the "nbf" claim defined in Section 4.1.5 of JWT [RFC7519], except that the value is of type NumericDate. The Claim Key 5 is used to identify this claim.

3.1.6. iat (Issued At) Claim

The "iat" (issued at) claim has the same meaning, syntax, and processing rules as the "iat" claim defined in Section 4.1.6 of JWT

[RFC7519], except that the value is of type NumericDate. The Claim Key 6 is used to identify this claim.

3.1.7. cti (CWT ID) Claim

The "cti" (CWT ID) claim has the same meaning, syntax, and processing rules as the "jti" claim defined in Section 4.1.7 of JWT [RFC7519], except that the value is of type binary string. The Claim Key 7 is used to identify this claim.

4. Summary of the claim names, keys, and value types

Name	Key	Value type
iss	1	text string
sub	2	text string
aud	3	text string
exp	4	integer or floating-point number
nbf	5	integer or floating-point number
iat	6	integer or floating-point number
cti	7	binary string

Figure 1: Summary of the claim names, keys, and value types

5. CBOR Tags and Claim Values

The claim values defined in this specification MUST NOT be prefixed with any CBOR tag. For instance, while CBOR tag 1 (epoch-based date/time) could logically be prefixed to values of the "exp", "nbf", and "iat" claims, this is unnecessary, since the representation of the claim values is already specified by the claim definitions. Tagging claim values would only take up extra space without adding information. However, this does not prohibit future claim definitions from requiring the use of CBOR tags for those specific claims.

6. CWT CBOR Tag

How to determine that a CBOR data structure is a CWT is application-dependent. In some cases, this information is known from the application context, such as from the position of the CWT in a data structure at which the value must be a CWT. One method of indicating that a CBOR object is a CWT is the use of the "application/cwt" content type by a transport protocol.

This section defines the CWT CBOR tag as another means for applications to declare that a CBOR data structure is a CWT. Its use is optional, and is intended for use in cases in which this information would not otherwise be known.

If present, the CWT tag MUST prefix a tagged object using one of the COSE CBOR tags. In this example, the COSE_Mac0 tag is used. The actual COSE_Mac0 object has been excluded from this example.

```
/ CWT CBOR tag / 61(
  / COSE_Mac0 CBOR tag / 17(
    / COSE_Mac0 object /
  )
)
```

Figure 2: Example of a CWT tag usage

7. Creating and Validating CWTs

7.1. Creating a CWT

To create a CWT, the following steps are performed. The order of the steps is not significant in cases where there are no dependencies between the inputs and outputs of the steps.

1. Create a CWT Claims Set containing the desired claims.

2. Let the Message be the binary representation of the CWT Claims Set.

3. Create a COSE Header containing the desired set of Header Parameters. The COSE Header MUST be valid per the [I-D.ietf-cose-msg] specification.

4. Depending upon whether the CWT is signed, MACed, or encrypted, there are three cases:

   * If the CWT is signed, create a COSE_Sign/COSE_Sign1 object using the Message as the COSE_Sign/COSE_Sign1 Payload; all steps specified in [I-D.ietf-cose-msg] for creating a COSE_Sign/COSE_Sign1 object MUST be followed.

   * Else, if the CWT is MACed, create a COSE_Mac/COSE_Mac0 object using the Message as the COSE_Mac/COSE_Mac0 Payload; all steps specified in [I-D.ietf-cose-msg] for creating a COSE_Mac/COSE_Mac0 object MUST be followed.

   * Else, if the CWT is a COSE_Encrypt/COSE_Encrypt0 object, create a COSE_Encrypt/COSE_Encrypt0 using the Message as the plaintext for the COSE_Encrypt/COSE_Encrypt0 object; all steps specified in [I-D.ietf-cose-msg] for creating a COSE_Encrypt/ COSE_Encrypt0 object MUST be followed.

5. If a nested signing, MACing, or encryption operation will be performed, let the Message be the COSE_Sign/COSE_Sign1, COSE_Mac/ COSE_Mac0, or COSE_Encrypt/COSE_Encrypt0, add the matching COSE CBOR tag, and return to Step 3.

6. If needed by the application, add the appropriate COSE CBOR tag to the COSE object to indicate the type of the COSE object. If needed by the application, add the CWT CBOR tag to indicate that the COSE object is a CWT.

7.2. Validating a CWT

When validating a CWT, the following steps are performed. The order of the steps is not significant in cases where there are no dependencies between the inputs and outputs of the steps. If any of the listed steps fail, then the CWT MUST be rejected -- that is, treated by the application as invalid input.

1. Verify that the CWT is a valid CBOR object.

2. If the object begins with the CWT CBOR tag, remove it and verify that one of the COSE CBOR tags follows it.

3. If the object is tagged with one of the COSE CBOR tags, remove it and use it to determine the type of the CWT, COSE_Sign/ COSE_Sign1, COSE_Mac/COSE_Mac0, or COSE_Encrypt/COSE_Encrypt0. If the object does not have a COSE CBOR tag, the COSE message type is determined from the application context.

4. Verify that the resulting COSE Header includes only parameters and values whose syntax and semantics are both understood and supported or that are specified as being ignored when not understood.

5. Depending upon whether the CWT is a signed, MACed, or encrypted, there are three cases:

   * If the CWT is a COSE_Sign/COSE_Sign1, follow the steps specified in [I-D.ietf-cose-msg] Section 4 (Signing Objects) for validating a COSE_Sign/COSE_Sign1 object. Let the Message be the COSE_Sign/COSE_Sign1 payload.

* Else, if the CWT is a COSE_Mac/COSE_Mac0, follow the steps specified in [I-D.ietf-cose-msg] Section 6 (MAC Objects) for validating a COSE_Mac/COSE_Mac0 object. Let the Message be the COSE_Mac/COSE_Mac0 payload.

* Else, if the CWT is a COSE_Encrypt/COSE_Encrypt0 object, follow the steps specified in [I-D.ietf-cose-msg] Section 5 (Encryption Objects) for validating a COSE_Encrypt/ COSE_Encrypt0 object. Let the Message be the resulting plaintext.

6. If the Message begins with a COSE CBOR tag, then the Message is a CWT that was the subject of nested signing, MACing, or encryption operations. In this case, return to Step 1, using the Message as the CWT.

7. Verify that the Message is a valid CBOR map; let the CWT Claims Set be this CBOR map.

8. Security Considerations

The security of the CWT relies upon on the protections offered by COSE. Unless the claims in a CWT are protected, an adversary can modify, add, or remove claims.

Since the claims conveyed in a CWT may be used to make authorization decisions, it is not only important to protect the CWT in transit but also to ensure that the recipient can authenticate the party that assembled the claims and created the CWT. Without trust of the recipient in the party that created the CWT, no sensible authorization decision can be made. Furthermore, the creator of the CWT needs to carefully evaluate each claim value prior to including it in the CWT so that the recipient can be assured of the validity of the information provided.

While syntactically, the signing and encryption operations for Nested CWTs may be applied in any order, if both signing and encryption are necessary, normally producers should sign the message and then encrypt the result (thus encrypting the signature). This prevents attacks in which the signature is stripped, leaving just an encrypted message, as well as providing privacy for the signer. Furthermore, signatures over encrypted text are not considered valid in many jurisdictions.

9. IANA Considerations

9.1. CBOR Web Token (CWT) Claims Registry

This section establishes the IANA "CBOR Web Token (CWT) Claims" registry.

Values are registered on a Specification Required [RFC5226] basis after a three-week review period on the cwt-reg-review@ietf.org mailing list, on the advice of one or more Designated Experts. However, to allow for the allocation of values prior to publication, the Designated Experts may approve registration once they are satisfied that such a specification will be published. [[Note to the RFC Editor: The name of the mailing list should be determined in consultation with the IESG and IANA. Suggested name: cwt-reg-review@ietf.org.]]

Registration requests sent to the mailing list for review should use an appropriate subject (e.g., "Request to register claim: example"). Registration requests that are undetermined for a period longer than 21 days can be brought to the IESG's attention (using the iesg@ietf.org mailing list) for resolution.

Criteria that should be applied by the Designated Experts includes determining whether the proposed registration duplicates existing functionality, whether it is likely to be of general applicability or whether it is useful only for a single application, and whether the registration description is clear.

It is suggested that multiple Designated Experts be appointed who are able to represent the perspectives of different applications using this specification in order to enable broadly informed review of registration decisions. In cases where a registration decision could be perceived as creating a conflict of interest for a particular Expert, that Expert should defer to the judgment of the other Experts.

9.1.1. Registration Template

Claim Name:
The human-readable name requested (e.g., "iss").

Claim Description:
Brief description of the claim (e.g., "Issuer").

JWT Claim Name:
Claim Name of the equivalent JWT claim, as registered in [IANA.JWT.Claims]. CWT claims should normally have a

corresponding JWT claim. If a corresponding JWT claim would not make sense, the Designated Experts can choose to accept registrations for which the JWT Claim Name is listed as "N/A".

Claim Key:
CBOR map key for the claim. Integer values between -256 and 255 and strings of length 1 are designated as Standards Track Document required. Integer values from -65536 to 65535 and strings of length 2 are designated as Specification Required. Integer values of greater than 65535 and strings of length greater than 2 are designated as expert review. Integer values less than -65536 are marked as private use.

Claim Value Type(s):
CBOR types that can be used for the claim value.

Change Controller:
For Standards Track RFCs, list the "IESG". For others, give the name of the responsible party. Other details (e.g., postal address, email address, home page URI) may also be included.

Specification Document(s):
Reference to the document or documents that specify the parameter, preferably including URIs that can be used to retrieve copies of the documents. An indication of the relevant sections may also be included but is not required.

9.1.2. Initial Registry Contents

- Claim Name: (RESERVED)
- Claim Description: This registration reserves the key value 0.
- JWT Claim Name: N/A
- Claim Key: 0
- Claim Value Type(s): N/A
- Change Controller: IESG
- Specification Document(s): [[this specification]]

- Claim Name: "iss"
- Claim Description: Issuer
- JWT Claim Name: "iss"
- Claim Key: 1
- Claim Value Type(s): text string
- Change Controller: IESG
- Specification Document(s): Section 3.1.1 of [[this specification]]

- Claim Name: "sub"
- Claim Description: Subject

- JWT Claim Name: "sub"
- Claim Key: 2
- Claim Value Type(s): text string
- Change Controller: IESG
- Specification Document(s): Section 3.1.2 of [[this specification]]

- Claim Name: "aud"
- Claim Description: Audience
- JWT Claim Name: "aud"
- Claim Key: 3
- Claim Value Type(s): text string
- Change Controller: IESG
- Specification Document(s): Section 3.1.3 of [[this specification]]

- Claim Name: "exp"
- Claim Description: Expiration Time
- JWT Claim Name: "exp"
- Claim Key: 4
- Claim Value Type(s): integer or floating-point number
- Change Controller: IESG
- Specification Document(s): Section 3.1.4 of [[this specification]]

- Claim Name: "nbf"
- Claim Description: Not Before
- JWT Claim Name: "nbf"
- Claim Key: 5
- Claim Value Type(s): integer or floating-point number
- Change Controller: IESG
- Specification Document(s): Section 3.1.5 of [[this specification]]

- Claim Name: "iat"
- Claim Description: Issued At
- JWT Claim Name: "iat"
- Claim Key: 6
- Claim Value Type(s): integer or floating-point number
- Change Controller: IESG
- Specification Document(s): Section 3.1.6 of [[this specification]]

- Claim Name: "cti"
- Claim Description: CWT ID
- JWT Claim Name: "jti"
- Claim Key: 7
- Claim Value Type(s): binary string

- Change Controller: IESG
- Specification Document(s): Section 3.1.7 of [[this specification]]

9.2. Media Type Registration

This section registers the "application/cwt" media type in the "Media Types" registry [IANA.MediaTypes] in the manner described in RFC 6838 [RFC6838], which can be used to indicate that the content is a CWT.

9.2.1. Registry Contents

- Type name: application
- Subtype name: cwt
- Required parameters: N/A
- Optional parameters: N/A
- Encoding considerations: binary
- Security considerations: See the Security Considerations section of [[this specification]]
- Interoperability considerations: N/A
- Published specification: [[this specification]]
- Applications that use this media type: IoT applications sending security tokens over HTTP(S) and other transports.
- Fragment identifier considerations: N/A
- Additional information:

  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, iesg@ietf.org
- Intended usage: COMMON
- Restrictions on usage: none
- Author: Michael B. Jones, mbj@microsoft.com
- Change controller: IESG
- Provisional registration? No

9.3. CoAP Content-Formats Registration

This section registers the CoAP Content-Format ID for the "application/cwt" media type in the "CoAP Content-Formats" registry [IANA.CoAP.Content-Formats].

9.3.1. Registry Contents

- Media Type: application/cwt
- Encoding: -
- Id: TBD (maybe 61)
- Reference: [[this specification]]

9.4. CBOR Tag registration

This section registers the CWT CBOR tag in the "CBOR Tags" registry [IANA.CBOR.Tags].

9.4.1. Registry Contents

- CBOR Tag: TBD (maybe 61 to use the same value as the Content-Format)
- Data Item: CBOR Web Token (CWT)
- Semantics: CBOR Web Token (CWT), as defined in [[this specification]]
- Reference: [[this specification]]
- Point of Contact: Michael B. Jones, mbj@microsoft.com

10. References

10.1. Normative References

[I-D.ietf-cose-msg]
Schaad, J., "CBOR Object Signing and Encryption (COSE)", draft-ietf-cose-msg-24 (work in progress), November 2016.

[IANA.CBOR.Tags]
IANA, "Concise Binary Object Representation (CBOR) Tags", <http://www.iana.org/assignments/cbor-tags/cbor-tags.xhtml>.

[IANA.CoAP.Content-Formats]
IANA, "CoAP Content-Formats", <http://www.iana.org/assignments/core-parameters/core-parameters.xhtml#content-formats>.

[IANA.MediaTypes]
IANA, "Media Types", <http://www.iana.org/assignments/media-types>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC7049] Bormann, C. and P. Hoffman, "Concise Binary Object Representation (CBOR)", RFC 7049, DOI 10.17487/RFC7049, October 2013, <http://www.rfc-editor.org/info/rfc7049>.

[RFC7519] Jones, M., Bradley, J., and N. Sakimura, "JSON Web Token (JWT)", RFC 7519, DOI 10.17487/RFC7519, May 2015, <http://www.rfc-editor.org/info/rfc7519>.

10.2. Informative References

[IANA.JWT.Claims] IANA, "JSON Web Token Claims", <http://www.iana.org/assignments/jwt>.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", RFC 5226, DOI 10.17487/RFC5226, May 2008, <http://www.rfc-editor.org/info/rfc5226>.

[RFC6838] Freed, N., Klensin, J., and T. Hansen, "Media Type Specifications and Registration Procedures", BCP 13, RFC 6838, DOI 10.17487/RFC6838, January 2013, <http://www.rfc-editor.org/info/rfc6838>.

[RFC7515] Jones, M., Bradley, J., and N. Sakimura, "JSON Web Signature (JWS)", RFC 7515, DOI 10.17487/RFC7515, May 2015, <http://www.rfc-editor.org/info/rfc7515>.

[RFC7516] Jones, M. and J. Hildebrand, "JSON Web Encryption (JWE)", RFC 7516, DOI 10.17487/RFC7516, May 2015, <http://www.rfc-editor.org/info/rfc7516>.

Appendix A. Examples

This appendix includes a set of CWT examples that show how the CWT Claims Set can be protected. There are examples that are signed, MACed, encrypted, and that use nested signing and encryption. To make the examples easier to read, they are presented both as hex strings and in the extended CBOR diagnostic notation described in Section 6 of [RFC7049].

A.1. Example CWT Claims Set

The CWT Claims Set used for the different examples displays usage of all the defined claims. For signed and MACed examples, the CWT Claims Set is the CBOR encoding as a binary string.

```
a70175636f61703a2f2f61732e6578616d706c652e636f6d02656572696b7703
7818636f61703a2f2f6c696768742e6578616d706c652e636f6d041a5612aeb0
051a5610d9f0061a5610d9f007420b71
```

Figure 3: Example CWT Claims Set as hex string

```
{
  / iss / 1: "coap://as.example.com",
  / sub / 2: "erikw",
  / aud / 3: "coap://light.example.com",
  / exp / 4: 1444064944,
  / nbf / 5: 1443944944,
  / iat / 6: 1443944944,
  / cti / 7: h'0b71'
}
```

Figure 4: Example CWT Claims Set in CBOR diagnostic notation

A.2. Example keys

This section contains the keys used to sign, MAC, and encrypt the messages in this appendix. Line breaks are for display purposes only.

A.2.1. 128-bit Symmetric Key as Hex Encoded String

```
231f4c4d4d3051fdc2ec0a3851d5b383
```

A.2.2. 256-bit Symmetric Key as Hex Encoded String

```
403697de87af64611c1d32a05dab0fe1fcb715a86ab435f1ec99192d79569388
```

A.2.3. ECDSA P-256 256-bit COSE Key

```
a622582060f7f1a780d8a783bfb7a2dd6b2796e8128dbbcef9d3d168db952997
1a36e7b92358206c1382765aec5358f117733d281c1c7bdc39884d04a45a1e6c
67c858bc206c19032601022158201432 9cce7868e416927599cf65a34f3ce2f
fda55a7eca69ed8919a394d42f0f2001
```

Figure 5: ECDSA 256-bit COSE Key as hex string

```
{
  / d /   -4: h'6c1382765aec5358f117733d281c1c7bdc39884d04a45a1e
                6c67c858bc206c19',
  / y /   -3: h'60f7f1a780d8a783bfb7a2dd6b2796e8128dbbcef9d3d168
                db9529971a36e7b9',
  / x /   -2: h'143329cce7868e416927599cf65a34f3ce2ffda55a7eca69
                ed8919a394d42f0f',
  / crv / -1: 1 / P-256 / ,
  / kty /  1: 2 / EC2 / ,
  / alg /  3: -7 / ECDSA 256 /
}
```

Figure 6: ECDSA 256-bit COSE Key in CBOR diagnostic notation

A.3. Example Signed CWT

This section shows a signed CWT with a single recipient and a full CWT Claims Set.

The signature is generated using the private key listed in Appendix A.2.3 and it can be validated using the public key from Appendix A.2.3. Line breaks are for display purposes only.

```
d28443a10126a05850a70175636f61703a2f2f61732e6578616d706c652e636f6
d02656572696b77037818636f61703a2f2f6c696768742e6578616d706c652e63
6f6d041a5612aeb0051a5610d9f0061a5610d9f007420b715840b9b2821b6b2c2
f9d1d984b11854dcfcee1f219746800ce76112c21f58c45dea1d7f01cec1ab394
0f75c459305365210a23a9ed463b4f6fc984c2f1c08e504d90
```

Figure 7: Signed CWT as hex string

```
18(
  [
    / protected / h'a10126' / {
      / alg / 1: -7 / ECDSA 256 /
    } / ,
    / unprotected / {},
    / payload / h'a70175636f61703a2f2f61732e6578616d706c652e63
                 6f6d02656572696b77037818636f61703a2f2f6c6967
                 68742e6578616d706c652e636f6d041a5612aeb0051a
                 5610d9f0061a5610d9f007420b71' / {
       / iss / 1: "coap://as.example.com",
       / sub / 2: "erikw",
       / aud / 3: "coap://light.example.com",
       / exp / 4: 1444064944,
       / nbf / 5: 1443944944,
       / iat / 6: 1443944944,
       / cti / 7: h'0b71'
      } / ,
    / signature / h'b9b2821b6b2c2f9d1d984b11854dcfcee1f2197468
                   00ce76112c21f58c45dea1d7f01cec1ab3940f75c4
                   59305365210a23a9ed463b4f6fc984c2f1c08e504d
                   90'
  ]
)
```

Figure 8: Signed CWT in CBOR diagnostic notation

A.4. Example MACed CWT

This section shows a MACed CWT with a single recipient, a full CWT Claims Set, and a CWT tag.

The MAC is generated using the 256-bit symmetric key from Appendix A.2.2 with a 64-bit truncation. Line breaks are for display purposes only.

```
d83dd18443a10104a05850a70175636f61703a2f2f61732e6578616d706c652e
636f6d02656572696b77037818636f61703a2f2f6c696768742e6578616d706c
652e636f6d041a5612aeb0051a5610d9f0061a5610d9f007420b7148093101ef
6d789200
```

Figure 9: MACed CWT with CWT tag as hex string

```
61(
  17(
    [
      / protected / h'a10104' / {
        / alg / 1: 4 / HMAC 256/64 /
      } / ,
      / unprotected / {},
      / payload / h'a70175636f61703a2f2f61732e6578616d706c652e636f
                   6d02656572696b77037818636f61703a2f2f6c69676874
                   2e6578616d706c652e636f6d041a5612aeb0051a5610d9
                   f0061a5610d9f007420b71' / {
          / iss / 1: "coap://as.example.com",
          / sub / 2: "erikw",
          / aud / 3: "coap://light.example.com",
          / exp / 4: 1444064944,
          / nbf / 5: 1443944944,
          / iat / 6: 1443944944,
          / cti / 7: h'0b71'
        } / ,
      / tag / h'093101ef6d789200'
    ]
  )
)
```

Figure 10: MACed CWT with CWT tag in CBOR diagnostic notation

A.5. Example Encrypted CWT

This section shows an encrypted CWT with a single recipient and a full CWT Claims Set.

The encryption is done with AES-CCM mode using the 128-bit symmetric key from Appendix A.2.1 with a 64-bit tag and 13-byte nonce, i.e., COSE AES-CCM-16-64-128. Line breaks are for display purposes only.

```
d08343a1010aa1054d3d9624bfb90a612bdcfc5077c45858e06d4b57cf3b3c9d
a3a16325dadcb9d2a0748f00ecd728f4b79030b56a292ee9cc8cc75349c120fc
1ba5d67ee29affde28df75a20f344812453ff68270ad5f46295660558168e1d1
85cb308226cdad0a50417dcd4a8d4b47
```

Figure 11: Encrypted CWT as hex string

```
16(
  [
    / protected / h'a1010a' / {
      / alg / 1: 10 / AES-CCM-16-64-128 /
    } /,
    / unprotected / {
      / iv / 5: h'3d9624bfb90a612bdcfc5077c4'
    },
    / ciphertext / h'e06d4b57cf3b3c9da3a16325dadcb9d2a0748f00ecd
                     728f4b79030b56a292ee9cc8cc75349c120fc1ba5d6
                     7ee29affde28df75a20f344812453ff68270ad5f462
                     95660558168e1d185cb308226cdad0a50417dcd4a8d
                     4b47'
  ]
)
```

Figure 12: Encrypted CWT in CBOR diagnostic notation

A.6. Example Nested CWT

This section shows a Nested CWT, signed and then encrypted, with a single recipient and a full CWT Claims Set.

The signature is generated using the private ECDSA key from Appendix A.2.3 and it can be validated using the public ECDSA parts from Appendix A.2.3. The encryption is done with AES-CCM mode using the 128-bit symmetric key from Appendix A.2.1 with a 64-bit tag and 13-byte nonce, i.e., COSE AES-CCM-16-64-128. The content type is set to CWT to indicate that there are multiple layers of COSE protection before finding the CWT Claims Set. The decrypted ciphertext will be a COSE_sign1 structure. In this example, it is the same one as in Appendix A.3, i.e., a Signed CWT Claims Set. Note that there is no limitation to the number of layers; this is an example with two layers. Line breaks are for display purposes only.

```
d08343a1010aa1054dd3bdeeb4daaa50625a5b576cc458a3318af5c80a11e081
91ca790b0793156451afc144e0f9f892679dff1d01cd52d7fe1e43ac8dabace0
f74af095f918197da1550a76d59c2a89db6331e12451fc87fef56f2ff179fb33
d6132ca34eb7fa8de0960d5f02a2b625792ccc8e5b3d59c0bede9d7438dc5c4f
e0c403c8fc32e874fbb7516c52edddfc09d4444a762dcd0cd486895131c343ae
040620cdd4448c6ce0b7803022ff3d7877a83c345c05a57b36105a
```

Figure 13: Signed and Encrypted CWT as hex string

```
16(
  [
    / protected / h'a203183d010a' / {
      / alg / 1: 10 / AES-CCM-16-64-128 /
    } / ,
    / unprotected / {
      / iv / 5: h'd3bdeeb4daaa50625a5b576cc4'
    },
    / ciphertext / h'318af5c80a11e08191ca790b0793156451afc144e0f
                     9f892679dff1d01cd52d7fe1e43ac8dabace0f74af0
                     95f918197da1550a76d59c2a89db6331e12451fc87f
                     ef56f2ff179fb33d6132ca34eb7fa8de0960d5f02a2
                     b625792ccc8e5b3d59c0bede9d7438dc5c4fe0c403c
                     8fc32e874fbb7516c52edddfc09d4444a762dcd0cd4
                     86895131c343ae040620cdd4448c6ce0b7803022ff3
                     d7877a83c345c05a57b36105a'
  ]
)
```

Figure 14: Signed and Encrypted CWT in CBOR diagnostic notation

A.7. Example MACed CWT with a floating-point value

This section shows a MACed CWT with a single recipient and a simple CWT Claims Set. The CWT Claims Set with a floating-point 'iat' value.

The MAC is generated using the 256-bit symmetric key from Appendix A.2.2 with a 64-bit truncation. Line breaks are for display purposes only.

```
d18443a10104a04ba106fb41d584367c20000048b8816f34c0542892
```

Figure 15: MACed CWT with a floating-point value as hex string

```
17(
  [
    / protected / h'a10104' / {
      / alg / 1: 4 / HMAC 256/64 /
    } / ,
    / unprotected / {},
    / payload / h'a106fb41d584367c200000' / {
        / iat / 6: 1443944944.5
      } / ,
    / tag / h'b8816f34c0542892'
  ]
)
```

Figure 16: MACed CWT with a floating-point value in CBOR diagnostic notation

Appendix B. Acknowledgements

This specification is based on JSON Web Token (JWT) [RFC7519], the authors of which also include Nat Sakimura and John Bradley. It also incorporates suggestions made by many people, notably Carsten Bormann, Jim Schaad, Ludwig Seitz, and Goeran Selander.

Appendix C. Document History

[[to be removed by the RFC Editor before publication as an RFC]]

-06

- Addressed review comments by Carsten Bormann and Jim Schaad. All changes were editorial in nature.

-05

- Addressed working group last call comments with the following changes:
- Say that CWT is derived from JWT, rather than CWT is a profile of JWT.
- Used CBOR type names in descriptions, rather than major/minor type numbers.
- Clarified the NumericDate and StringOrURI descriptions.
- Changed to allow CWT claim names to use values of any legal CBOR map key type.

- Changed to use the CWT tag to identify nested CWTs instead of the CWT content type.
- Added an example using a floating-point date value.
- Acknowledged reviewers.

-04

- Specified that the use of CBOR tags to prefix any of the claim values defined in this specification is NOT RECOMMENDED.

-03

- Reworked the examples to include signed, MACed, encrypted, and nested CWTs.
- Defined the CWT CBOR tag and explained its usage.

-02

- Added IANA registration for the application/cwt media type.
- Clarified the nested CWT language.
- Corrected nits identified by Ludwig Seitz.

-01

- Added IANA registration for CWT Claims.
- Added IANA registration for the application/cwt CoAP content-format type.
- Added Samuel Erdtman as an editor.
- Changed Erik's e-mail address.

-00

- Created the initial working group version based on draft-wahlstroem-ace-cbor-web-token-00.

Authors' Addresses

Michael B. Jones
Microsoft

Email: mbj@microsoft.com
URI: http://self-issued.info/

Erik Wahlstroem
Sweden

Email: erik@wahlstromstekniska.se

Samuel Erdtman
Spotify AB
Birger Jarlsgatan 61, 4tr
Stockholm 113 56
Sweden

Phone: +46702691499
Email: erdtman@spotify.com

Hannes Tschofenig
ARM Ltd.
Hall in Tirol 6060
Austria

Email: Hannes.Tschofenig@arm.com