

ACE Working Group
Internet-Draft
Intended status: Standards Track
Expires: September 3, 2017

M. Jones
Microsoft
E. Wahlstroem

S. Erdtman
Spotify AB
H. Tschofenig
ARM Ltd.
March 2, 2017

CBOR Web Token (CWT)
draft-ietf-ace-cbor-web-token-03

Abstract

CBOR Web Token (CWT) is a compact means of representing claims to be transferred between two parties. CWT is a profile of the JSON Web Token (JWT) that is optimized for constrained devices. 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.

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 September 3, 2017.

Copyright Notice

Copyright (c) 2017 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 (<http://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.	Terminology	3
3.	Claims	4
3.1.	Claim Names	4
3.1.1.	iss (Issuer) Claim	4
3.1.2.	sub (Subject) Claim	4
3.1.3.	aud (Audience) Claim	5
3.1.4.	exp (Expiration Time) Claim	5
3.1.5.	nbft (Not Before) Claim	5
3.1.6.	iat (Issued At) Claim	5
3.1.7.	cti (CWT ID) Claim	5
4.	Summary of the values, CBOR major types and encoded claim keys	5
5.	CWT CBOR Tag	6
6.	Creating and Validating CWTs	6
6.1.	Creating a CWT	6
6.2.	Validating a CWT	7
7.	Security Considerations	9
8.	IANA Considerations	9
8.1.	CBOR Web Token (CWT) Claims Registry	9
8.1.1.	Registration Template	9
8.1.2.	Initial Registry Contents	10
8.2.	Media Type Registration	11
8.2.1.	Registry Contents	11
8.3.	CoAP Content-Formats Registration	12
8.3.1.	Registry Contents	12
8.4.	CBOR Tag registration	12
8.4.1.	Registry Contents	12
9.	References	13
9.1.	Normative References	13
9.2.	Informative References	14
Appendix A.	Examples	14
A.1.	Example CWT Claims Set	14
A.2.	Example keys	15
A.2.1.	128-bit Symmetric Key as Hex Encoded String	15
A.2.2.	256-bit Symmetric Key as Hex Encoded String	15

A.2.3.	ECDSA P-256 256-bit COSE Key	15
A.3.	Example Signed CWT	16
A.4.	Example MACed CWT	17
A.5.	Example Encrypted CWT	18
A.6.	Example Nested CWT	19
Appendix B.	Acknowledgements	20
Appendix C.	Document History	20
	Authors' Addresses	21

[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 Signatures (JWS) [[RFC7515](#)] and JSON Web Encryption (JWE) [[RFC7516](#)] to secure the contents of the JWT, which is a set of claims represented in JSON [[RFC7519](#)]. 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.

In this document an alternative encoding of claims is defined. 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".

[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](#)].

Type3StringOrURI:

The "Type3StringOrURI" term has the same meaning, syntax, and processing rules as the "StringOrUri" term defined in [Section 2](#) of

JWT [[RFC7519](#)], except that Type3StringOrURI uses CBOR major type 3 instead of a JSON string value.

Type6NumericDate:

The "Type6NumericDate" term has the same meaning, syntax, and processing rules as the "NumericDate" term defined in [Section 2](#) of JWT [[RFC7519](#)], except that Type6NumericDate uses CBOR major type 6, with tag value 1, instead of a numeric JSON value.

CBOR encoded claim key:

The key used to identify a claim value.

CWT Claims Set

A 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 CBOR encoded claim keys are represented using CBOR major type 0. [Section 4](#) summarizes all keys used to identify the claims defined in this document.

[3.1. Claim Names](#)

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 format MUST be a Type3StringOrURI. The CBOR encoded claim key 1 MUST be 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 format MUST be a Type3StringOrURI. The CBOR encoded claim key 2 MUST be 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 format MUST be a Type3StringOrURI. The CBOR encoded claim key 3 MUST be 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 format MUST be a Type6NumericDate. The CBOR encoded claim key 4 MUST be 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 format MUST be a Type6NumericDate. The CBOR encoded claim key 5 MUST be 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 format MUST be a Type6NumericDate. The CBOR encoded claim key 6 MUST be 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 format MUST be of major type 2, binary string. The CBOR encoded claim key 7 MUST be used to identify this claim.

[4.](#) Summary of the values, CBOR major types and encoded claim keys

Claim	CBOR encoded claim key	CBOR major type of value
iss	1	3
sub	2	3
aud	3	3
exp	4	6 tag value 1
nbf	5	6 tag value 1
iat	6	6 tag value 1
cti	7	2

Figure 1: Summary of the values, CBOR major types and encoded claim keys.

5. 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.

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

6. Creating and Validating CWTs

6.1. 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, and return to Step 3, using a "content type" header value corresponding to the media type "application/cwt" in the new COSE Header created in that step.

Note: If integrity (signing/MACing) and confidentiality (encryption) protection are needed, it is recommended to use an authenticated encryption algorithm to save space and processing.
6. If needed by the application, add the appropriate COSE CBOR tag to the COSE object to indicate type of COSE object. If also needed by the application, add the CWT CBOR tag to indicate that the COSE object is a CWT.

[6.2.](#) 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 an 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 verify that it corresponds to the structure of the following COSE object.
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. Use the CBOR tag to determine the type of the CWT, COSE_Sign/COSE_Sign1, COSE_Mac/COSE_Mac0, or COSE_Encrypt/COSE_Encrypt0.
6. 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.
7. If the COSE Header contains a "content type" header value corresponding to the media type "application/cwt", then the Message is a CWT that was the subject of nested signing or encryption operations. In this case, return to Step 1, using the Message as the CWT.
8. Verify that the Message is a valid CBOR object; let the CWT Claims Set be this CBOR object.

7. Security Considerations

The security of the CWT is dependent 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.

8. IANA Considerations

8.1. 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, 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.

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.

8.1.1. 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".

CBOR Key Value:

Key value for the claim. The key value **MUST** be an integer in the range of 1 to 65536.

CBOR Major Type:

CBOR major type and optional tag for the claim.

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.

8.1.2. Initial Registry Contents

- o Claim Name: "iss"
- o Claim Description: Issuer
- o JWT Claim Name: "iss"
- o CBOR Key Value: 1
- o CBOR Major Type: 3
- o Change Controller: IESG
- o Specification Document(s): [Section 3.1.1](#) of [[this specification]]

- o Claim Name: "sub"
- o Claim Description: Subject
- o JWT Claim Name: "sub"
- o CBOR Key Value: 2
- o CBOR Major Type: 3
- o Change Controller: IESG
- o Specification Document(s): [Section 3.1.2](#) of [[this specification]]

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

- o Claim Name: "exp"

- o Claim Description: Expiration Time
- o JWT Claim Name: "exp"
- o CBOR Key Value: 4
- o CBOR Major Type: 6, tag value 1
- o Change Controller: IESG
- o Specification Document(s): [Section 3.1.4](#) of [[this specification]]
- o Claim Name: "nbf"
- o Claim Description: Not Before
- o JWT Claim Name: "nbf"
- o CBOR Key Value: 5
- o CBOR Major Type: 6, tag value 1
- o Change Controller: IESG
- o Specification Document(s): [Section 3.1.5](#) of [[this specification]]
- o Claim Name: "iat"
- o Claim Description: Issued At
- o JWT Claim Name: "iat"
- o CBOR Key Value: 6
- o CBOR Major Type: 6, tag value 1
- o Change Controller: IESG
- o Specification Document(s): [Section 3.1.6](#) of [[this specification]]
- o Claim Name: "cti"
- o Claim Description: CWT ID
- o JWT Claim Name: "jti"
- o CBOR Key Value: 7
- o CBOR Major Type: 2
- o Change Controller: IESG
- o Specification Document(s): [Section 3.1.7](#) of [[this specification]]

[8.2.](#) Media Type Registration

This section registers the "application/cwt" media type [[RFC2046](#)] 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.

[8.2.1.](#) Registry Contents

- o Type name: application
- o Subtype name: cwt
- o Required parameters: N/A
- o Optional parameters: N/A

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

Magic number(s): N/A

File extension(s): N/A

Macintosh file type code(s): N/A

- o Person & email address to contact for further information: IESG, iesg@ietf.org
- o Intended usage: COMMON
- o Restrictions on usage: none
- o Author: Michael B. Jones, mbj@microsoft.com
- o Change controller: IESG
- o Provisional registration? No

8.3. 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](#)] established by [[RFC7252](#)].

8.3.1. Registry Contents

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

8.4. CBOR Tag registration

This section registers the CWT CBOR tag in the "CBOR Tags" registry [[IANA.CBOR.Tags](#)] established by [[RFC7049](#)].

8.4.1. Registry Contents

- o CBOR Tag: TBD (maybe 61 to use the same value as the Content-Format)
- o Data Item: CBOR Web Token (CWT)
- o Semantics: CBOR Web Token (CWT), as defined in [[this specification]]
- o Reference: [[this specification]]

- o Point of Contact: Michael B. Jones, mbj@microsoft.com

9. References

9.1. 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.JWT.Claims]
IANA, "JSON Web Token Claims",
<<http://www.iana.org/assignments/jwt>>.
- [IANA.MediaTypees]
IANA, "Media Types",
<<http://www.iana.org/assignments/media-types>>.
- [RFC2046] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types", [RFC 2046](#), DOI 10.17487/RFC2046, November 1996,
<<http://www.rfc-editor.org/info/rfc2046>>.
- [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>>.
- [RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", [BCP 26](#), [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>>.

- [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>>.
- [RFC7159] Bray, T., Ed., "The JavaScript Object Notation (JSON) Data Interchange Format", [RFC 7159](#), DOI 10.17487/RFC7159, March 2014, <<http://www.rfc-editor.org/info/rfc7159>>.
- [RFC7252] Shelby, Z., Hartke, K., and C. Bormann, "The Constrained Application Protocol (CoAP)", [RFC 7252](#), DOI 10.17487/RFC7252, June 2014, <<http://www.rfc-editor.org/info/rfc7252>>.
- [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>>.
- [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>>.

9.2. Informative References

- [I-D.greevenbosch-appsawg-cbor-cddl]
Vigano, C. and H. Birkholz, "CBOR data definition language (CDDL): a notational convention to express CBOR data structures", [draft-greevenbosch-appsawg-cbor-cddl-09](#) (work in progress), September 2016.

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 [[I-D.greevenbosch-appsawg-cbor-cddl](#)].

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.


```
a702656572696b77037818636f61703a2f2f6c696768742e6578616d706c652e
636f6d041a5612aeb0051a5610d9f0061a5610d9f00175636f61703a2f2f6173
2e6578616d706c652e636f6d07420b71
```

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

```
9e4f3e65cc1a558b39ce97b3db469b04
```

[A.2.2.](#) 256-bit Symmetric Key as Hex Encoded String

```
e60198ac1650ec9210d7f4f5b27aeae2ada8f4adada555909edca75ce2ae506e
```

[A.2.3.](#) ECDSA P-256 256-bit COSE Key

```
a6225820feb11ca73b028a10cf77d58a2dfdf2a11eab8ffeeeaaeeb03097ffee
9f3ef2fc2358200657fada2568959c49a404583fe237290eb1956f3ad3d966
ea09e33369d7b103260102215820c4f9160fc22682991c59c4d96e8accc2da3c
c7b7a9bc197c7c1e1bc6d0c1dc612001
```

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


```
{
  / d / -4: h'0657fada2568959c49a404583fe237290eb1956f3ad3d966
    ea09e33369d7b1',
  / y / -3: h'feb11ca73b028a10cf77d58a2dfdf2a11eab8ffeeeaaeeb030
    97ffee9f3ef2fc',
  / x / -2: h'c4f9160fc22682991c59c4d96e8accc2da3cc7b7a9bc197c7c
    1e1bc6d0c1dc61',
  / 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 ECDSA key from [Appendix A.2.3](#) and it can be validated using the public part of the ECDSA key from [Appendix A.2.3](#). Line breaks are for display purposes only.

```
d28446a203183d0126a05850a702656572696b77037818636f61703a2f2f6c69
6768742e6578616d706c652e636f6d041a5612aeb0051a5610d9f0061a5610d9
f00175636f61703a2f2f61732e6578616d706c652e636f6d07420b7158407eef
29abe962ac185e5a372d95d69ce1b5683c5c25efb69a81710dc5173254f5179a
639827694c22828819704eb026676ca78aaf8da76672a6b5537fb90e710d
```

Figure 7: Signed CWT as hex string


```

18(
  [
    / protected / h'a203183d0126' / {
      / content type / 3: 61, / CWT /
      / alg / 1: -7 / ECDSA 256 /
    } / ,
    / unprotected / {},
    / payload / h'a702656572696b77037818636f61703a2f2f6c69676874
      2e6578616d706c652e636f6d041a5612aeb0051a5610d9
      f0061a5610d9f00175636f61703a2f2f61732e6578616d
      706c652e636f6d07420b71' / {
      / 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'7eef29abe962ac185e5a372d95d69ce1b5683c5c25ef
      b69a81710dc5173254f5179a639827694c2282881970
      4eb026676ca78aaf8da76672a6b5537fb90e710d'
  ]
)

```

Figure 8: Signed CWT in CBOR diagnostic notation

[A.4.](#) Example MACed CWT

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

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.

```

d18446a203183d0104a05850a702656572696b77037818636f61703a2f2f6c69
6768742e6578616d706c652e636f6d041a5612aeb0051a5610d9f0061a5610d9
f00175636f61703a2f2f61732e6578616d706c652e636f6d07420b7148b59884
6f1ce93f9d

```

Figure 9: MACed CWT as hex string


```

17(
  [
    / protected / h'a203183d0104' / {
      / content type / 3: 61, / CWT /
      / alg / 1: 4 / HMAC 256/64 /
    } / ,
    / unprotected / {},
    / payload / h'a702656572696b77037818636f61703a2f2f6c69676874
      2e6578616d706c652e636f6d041a5612aeb0051a5610d9
      f0061a5610d9f00175636f61703a2f2f61732e6578616d
      706c652e636f6d07420b71' / {
      / 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'b598846f1ce93f9d'
  ]
)

```

Figure 10: MACed CWT 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.

```

d08346a203183d010aa1054dadbe290e8c9c23067a558b15795858f7a8ec3e32
3bb6e006e8aec087666f6fc0d65d7aa272f5f1dde1dfb52fd3a5e1ace97e5bfc
8f05a146fd8a9feab7bb9e722254e2660612f956041264c06ea3b95afb0d8ce3
138bc80baf2511565d3dad63ea7534699fa449

```

Figure 11: Encrypted CWT as hex string


```

16(
  [
    / protected / h'a203183d010a' / {
      / content type / 3: 61, / CWT /
      / alg / 1: 10 / AES-CCM-16-64-128 /
    } /,
    / unprotected / {
      / iv / 5: h'adbe290e8c9c23067a558b1579'
    },
    / ciphertext / h'f7a8ec3e323bb6e006e8aec087666f6fc0d65d7aa27
      2f5f1dde1dfb52fd3a5e1ace97e5bfc8f05a146fd8a
      9feab7bb9e722254e2660612f956041264c06ea3b95
      afb0d8ce3138bc80baf2511565d3dad63ea7534699f
      a449'
  ]
)

```

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.

```

d08346a203183d010aa1054d2653469d58937647a6a1bb023458a65da538206c33
cf941df7ea933ba7b93c60322017f9db9c904608fce2688b51028b5b912f9010
ae72802bf65778593c7270b20683b1587824eb4074e03323ccf0541b495a3757
f353a8424b6ceeaec1898964d8a03e04e514a5b0ca143b57689a2a9f1c6c84d
535d1966adf900dfa0dd045d2325c40150a07d602b65c60e62894c870ad5fc2
cb709e4d17d381806797b6cf118608e18c3facd0a0ac09d88ea73d4ed7e3b57c

```

Figure 13: Signed and Encrypted CWT as hex string


```

16(
  [
    / protected / h'a203183d010a' / {
      / content type / 3: 61, / CWT /
      / alg / 1: 10 / AES-CCM-16-64-128 /
    } / ,
    / unprotected / {
      / iv / 5: h'2653469d58937647a6a1bb0234'
    },
    / ciphertext / h'5da538206c33cf941df7ea933ba7b93c60322017f9d
      b9c904608fce2688b51028b5b912f9010ae72802bf6
      5778593c7270b20683b1587824eb4074e03323ccf05
      41b495a3757f353a8424b6ceaaec1898964d8a03e0
      4e514a5b0ca143b57689a2a9f1c6c84d535d1966adf
      900dfaf0dd045d2325c40150a07d602b65c60e62894
      c870ad5fc2cb709e4d17d381806797b6cf118608e18
      c3facd0a0ac09d88ea73d4ed7e3b57c'
  ]
)

```

Figure 14: Signed and Encrypted CWT 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. Ludwig Seitz and Goeran Selander have made contributions the specification.

[Appendix C.](#) Document History

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

-03

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

-02

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

-01

- o Added IANA registration for CWT Claims.

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

-00

- o 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: erdman@spotify.com

Hannes Tschofenig
ARM Ltd.
Hall in Tirol 6060
Austria

Email: Hannes.Tschofenig@arm.com

