Network Working Group
Internet-Draft
Intended status: Informational
Expires: 2 September 2026

C. Bormann
Universit辰t Bremen TZI

1 March 2026

# External References to Values in CBOR Diagnostic Notation (EDN)
## draft-ietf-cbor-edn-e-ref-03

## Abstract

The Concise Binary Object Representation (CBOR, RFC 8949) is a data format whose design goals include the possibility of extremely small code size, fairly small message size, and extensibility without the need for version negotiation.

CBOR diagnostic notation (EDN) is widely used to represent CBOR data items in a way that is accessible to humans, for instance for examples in a specification. At the time of writing, EDN did not provide mechanisms for composition of such examples from multiple components or sources. This document uses EDN application extensions to provide two such mechanisms, both of which insert an imported data item into the data item being described in EDN:

The e'' application extension provides a way to import data items, particularly constant values, from a CDDL model (which itself has ways to provide composition).

The ref'' application extension provides a way to import data items that are described in EDN.

## About This Document

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

The latest revision of this draft can be found at https://cbor-wg.github.io/edn-e-ref/draft-ietf-cbor-edn-e-ref.html. Status information for this document may be found at https://datatracker.ietf.org/doc/draft-ietf-cbor-edn-e-ref/.

Discussion of this document takes place on the cbor Working Group mailing list (mailto:cbor@ietf.org), which is archived at https://mailarchive.ietf.org/arch/browse/cbor/. Subscribe at https://www.ietf.org/mailman/listinfo/cbor/.

Source for this draft and an issue tracker can be found at https://github.com/cbor-wg/edn-e-ref.

Status of This Memo



Copyright Notice



Table of Contents

## 1. Introduction

(Please see abstract.) [RFC8949] [I-D.ietf-cbor-edn-literals]

See [I-D.bormann-cbor-draft-numbers] for a more general discussion of working with assigned numbers during development of a specification.

## 2. The e'' application extension: importing from CDDL

### Problem

In diagnostic notation examples used during development of earlier drafts, authors often used text strings in place of constants they need, even though they actually mean a placeholder for a later, to-be-registered integer.

One example from a recent draft would be:

```
{
    "group_mode" : true,
    "gp_enc_alg" : 10,
          "hkdf" : 5
}
```

Figure 1: Misleading usage of text strings as stand-in for registered constants

Not only is the reader misled by seeing text strings in places that are actually intended to be small integers, there are also small integers that are not explained at all (here: 10, 5). The usefulness of this example is greatly reduced. Examples constructed in this are not actually machine-readable -- they seem to be, but they mean the wrong thing in several places without any warning that this is so.

### Solution

In many cases, the constants needed to clean up this example are already available in a CDDL model, or could be easily made available in this way.

If such a CDDL model can be identified, the EDN application extension e'constant-name' can be used to reference a constant defined by that model under the name constant-name. (Hint: memorize the e as external constant, or enum.)

For the example in Figure 1, such a CDDL model could have at least the content shown in Figure 2:

```
hkdf = -1
group_mode = -3
gp_enc_alg = -4
HMAC-256-256 = 5
AES-CCM-16-64-128 = 10
```

Figure 2: CDDL model defining constants for e''

Note that such a model can have other, unrelated CDDL rules that define more complex data items; only the ones used in an e'' construct need to be constant values.

Using the CDDL model in Figure 2, the example in Figure 1 can be notated as (example derived from Section 6.2-9 of [I-D.ietf-ace-oscore-gm-admin]):

```
{
    e'group_mode' : true,
    e'gp_enc_alg' : e'AES-CCM-16-64-128',
          e'hkdf' : e'HMAC-256-256'
}
```

Figure 3: Example updated to use e'constantname' for registered constants

This example is equivalent to notating {-3: true, -4: 10, -1: 5}, which expresses the concise 10-byte data item that will actually be interchanged for this example.

Note that the application-oriented literal does not itself define where the CDDL definitions it uses come from. This information needs to come from the context of the example.

Note also that the CDDL names for one EDN instance all come from a single name space. [I-D.bormann-cbor-edn-mapkey] proposes access to CDDL names where the name space in use is specific to the data item the reference has as immediate context; this provides a more complex, but also more powerful approach that can handle different constant values for the same name in different contexts.

## Implementation

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

The e'' application extension is now implemented in the cbor-diag tools [cbor-diag], by the cbor-diag-e gem [cbor-diag-e], which can be installed as:

```
 gem install cbor-diag-e cddlc
```

(cbor-diag-e uses cddlc [cddlc] internally, so it must be in PATH.)

Use of this extension has two prerequisites:

1. Opt-in to the application extension e, which in the cbor-diag tools such as diag2_x_.rb is done using the -a command line flag, here: -ae.

2. Identification of the CDDL model to be used, which will give the actual values for the constants.

   This can be a complete CDDL model for the application, no need to limit it just to constant definitions. (Where the constant values need to be obtained by registration at the time of completion of the document using the examples, the CDDL model can be set up with TBD values of the constants to be assigned, and once they are, the necessary updates are all in one place.)

Assuming that the example in Figure 3 is in a file called gmadmin.diag, and that the CDDL model that includes the constants defined in Figure 2 is in gmadmin.cddl, the following commands can be used to translate the e'' constants into their actual values:

```
$ export CBOR_DIAG_CDDL=gmadmin.cddl
$ diag2diag.rb -ae gmadmin.diag
{33: true, 34: 10, 31: 5}
```

## Provisional use

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

The need for this application is there now, while ratification of the present specification might take a year or so. Until then, each document using this scheme can simply use boilerplate such as:

> In the CBOR diagnostic notation used in this document, constructs of the form e' somename' are replaced by the value assigned to somename in CDDL in figure 0815. E.g., {e'group_mode': "bar"} stands for {33: "bar"}.

(Choose 0815, group_mode and 33 along the lines of the figure you include with the CDDL definitions needed.)

## 3. The ref'' application extension: importing from EDN

Problem

Examples using CBOR diagnostic notation can get large. One way to manage the size of an example is to make it incomplete. This reduces the usefulness of the example for machine-processing. It can also be misleading, unless the elision is made explicit (see Section 3.2 of [I-D.ietf-cbor-edn-literals]).

Solution

In a set of CBOR examples, recurring subtrees can often be identified, the details of which do not need to be repeated in each example.

By enabling examples to reference these subtrees from a separate piece of EDN, each example can focus on what is specific for them.

The ref'' application-oriented literal enables composition by standing for a CBOR data item from a separate EDN instance that is referenced using its text as an identifier.

So, for example, if 123.diag is a file containing

```
[1, 2, 3]
```

the result of the EDN

```
[4711.0, true, ref' 123.diag' ]
```

is

```
[4711.0, true, [1, 2, 3]]
```

The text of the literal can be one of two kinds of identifiers:

1. a file name to be interpreted in the context of the referencing example, as shown above, or

2. a URI that references the EDN to be embedded, as in

```
[4711.0, true, ref'http://tzi.de/~cabo/123.diag']
```

| (ISSUE: We could use upper-case REF to unambiguously identify one of he two; the current implementation however just tries to parse the literal text as a URI and, if that fails, interprets it as a file name.)

Note that a ref'' application-oriented literal can only be used for a single CBOR data item; the extension point provided by EDN does not work for splicing in CBOR sequences.

## Implementation

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

The ref'' application extension is now implemented in the cbor-diag tools [cbor-diag], by the cbor-diag-ref gem [cbor-diag-ref], which can be installed as:

```
gem install cbor-diag-ref
```

For using the application extension, the cbor-diag tools such as diag2_x_.rb need to be informed by the -a command line flag, here: -aref.

For experimenting with the implementation, the web resource http://tzi.de/~cabo/123.diag contains [1, 2, 3]. This example enables usage as in:

```
$ echo "[4711.0, true, ref'http://tzi.de/~cabo/123.diag']" >my.diag
$ diag2diag.rb -aref my.diag
[4711.0, true, [1, 2, 3]]

$ echo "[4, 5, 6]" >456.diag
$ echo "[4711.0, true, ref'456.diag']" >my.diag
$ diag2diag.rb -aref my.diag
[4711.0, true, [4, 5, 6]]
```

If a referenced EDN file parses as a CBOR sequence this is currently treated as an error.

## Provisional use

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

Documents that want to use the application extension ref'' now can use boilerplate similar to that given above for e''.

# 4. IANA Considerations

IANA is requested to make the following two assignments in the CBOR Diagnostic Notation Application-extension Identifiers registry [IANA.cbor-diagnostic-notation]:

| Application-extension Identifier | Description |
|---|---|
| e | import value from external CDDL |
| ref | import value from external EDN |

Table 1: Additions to Application-extension Identifier Registry

All entries the Change Controller "IETF" and the Reference "RFC-XXXX".

// RFC Editor: please replace RFC-XXXX with the RFC number of this
// RFC, [IANA.cbor-diagnostic-notation] with a reference to the new
// registry group, and remove this note.

## 5. Security considerations

The security considerations of [RFC8610], [RFC8949], and [I-D.bormann-t2trg-deref-id] apply.

The proof of concept implementations described do not do any sanitizing of URLs or file names at all. Upcoming versions of the present document will need to define the right restrictions for external references like this.

## 6. References

### 6.1. Normative References

[I-D.ietf-cbor-edn-literals]
Bormann, C., "CBOR Extended Diagnostic Notation (EDN)", Work in Progress, Internet-Draft, draft-ietf-cbor-edn-literals-19, 16 October 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-cbor-edn-literals-19>.

[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/rfc/rfc8610>.

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

## 6.2. Informative References

[cbor-diag] "CBOR diagnostic utilities", n.d., <https://github.com/cabo/cbor-diag>.

[cbor-diag-e] "CBOR diagnostic extension e’’", n.d., <https://github.com/cabo/cbor-diag-e>.

[cbor-diag-ref] "CBOR diagnostic extension ref’’", n.d., <https://github.com/cabo/cbor-diag-ref>.

[cddlc] "CDDL conversion utilities", n.d., <https://github.com/cabo/cddlc>.

[I-D.bormann-cbor-draft-numbers] Bormann, C., "Managing CBOR codepoints in Internet-Drafts", Work in Progress, Internet-Draft, draft-bormann-cbor-draft-numbers-07, 12 January 2026, <https://datatracker.ietf.org/doc/html/draft-bormann-cbor-draft-numbers-07>.

[I-D.bormann-cbor-edn-mapkey] Bormann, C. and C. Amsss, "CBOR: Generating Numeric Map Labels from Textual EDN", Work in Progress, Internet-Draft, draft-bormann-cbor-edn-mapkey-01, 1 March 2026, <https://datatracker.ietf.org/doc/html/draft-bormann-cbor-edn-mapkey-01>.

[I-D.bormann-t2trg-deref-id] Bormann, C. and C. Amsss, "The "dereferenceable identifier" pattern", Work in Progress, Internet-Draft, draft-bormann-t2trg-deref-id-07, 24 February 2026, <https://datatracker.ietf.org/doc/html/draft-bormann-t2trg-deref-id-07>.

[I-D.ietf-ace-oscore-gm-admin]
Tiloca, M., Hglund, R., Van der Stok, P., and F. Palombini, "Admin Interface for the OSCORE Group Manager", Work in Progress, Internet-Draft, draft-ietf-ace-oscore-gm-admin-15, 23 February 2026, <https://datatracker.ietf.org/doc/html/draft-ietf-ace-oscore-gm-admin-15>.

## Acknowledgements

TBD

## Author’s Address

Carsten Bormann
Universitt Bremen TZI
Postfach 330440
D-28359 Bremen
Germany
Phone: +49-421-218-63921
Email: cabo@tzi.org