Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: 17 October 2026

C. Bormann
Universit辰t Bremen TZI

15 April 2026

Additional Application Extensions for the CBOR Extended Diagnostic Notation (EDN)

draft-bormann-cbor-edn-app-ext-01

Abstract

The CBOR Extended Diagnostic Notation (EDN), to be standardized in draft-ietf-cbor-edn-literals, provides "application extensions" as its main language extension point.

A number of application extensions are already defined in draft-ietf-cbor-edn-literals itself and in draft-ietf-cbor-edn-e-ref. The present document defines a number of additional application extensions that have been batched up as a next step after completing these specifications. (
// Chore: Briefly List extensions.)

// This -01 of an individual submission is a slight update of -00,
// which showed the approximate shape the first "batch" of
// application extensions could have, plus a number of registrations
// that could go into this batch. The latter provides a basis for a
// technical discussion of those registrations.

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://cabo.github.io/edn-app-ext/draft-bormann-cbor-edn-app-ext.html. Status information for this document may be found at https://datatracker.ietf.org/doc/draft-bormann-cbor-edn-app-ext/.

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/cabo/edn-app-ext.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction 3
  1.1. Conventions and Terminology 3
2. Application Extensions 4
  2.1. same: multiple literals to be checked against each other 4
  2.2. bytes: Reinterpret text string as byte string 5
  2.3. float: IEEE754-oriented literals for more floating point values 5
  2.4. tbd b32/h32/c32?: Create byte string from base32 representation 6
  2.5. tbd b45: Create byte string from base45 representation . 6
3. Implementation Status 6
  3.1. Implementation 1 7
  3.2. Implementation 2 7
4. Security considerations 7

5. IANA considerations 7
6. References 7
6.1. Normative References 7
6.2. Informative References 9
List of Tables 9
Acknowledgements 9
Author's Address 9

1. Introduction

(See abstract.)

Name	Purpose
same	Multiple literals for the same item, to be checked against each other
bytes	Reinterpret text string as byte string
float	Provide IEEE754-oriented literals for more floating point values
tbd b32/h32/ c32?	Create byte string from base32 representation, possibly beyond the two variants defined in [RFC4648]
tbd b45	Create byte string from base45 representation [RFC9285]

Table 1: Additional EDN application extensions defined in this document

// Discuss: We should also add application extensions for text generation from bytes, such as b64c and b64u, along the lines of [RFC9741].

1.1. Conventions and Terminology

This specification uses terminology from [I-D.ietf-cbor-edn-literals]. In particular, with respect to control operators, "target" refers to the left-hand side operand, and "controller" to the right-hand side operand. "Tool" refers to tools that produce, consume, or otherwise process EDN. Note also that the data model underlying CBOR provides for text strings as well as byte

strings as two separate types, which are then collectively referred to as "strings".

The term ABNF in this specification stands for the combination of [STD68] and [RFC7405], i.e., ABNF defined in this document allows use of the case-sensitive extensions defined in [RFC7405].

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 [BCP14] (RFC2119) (RFC8174) when, and only when, they appear in all capitals, as shown here.

2. Application Extensions

A table like Table 6 in [I-D.ietf-cbor-edn-literals]: (Add text.)

app-prefix	content of single-quoted string	result type
same	sequence of one or more data items	data item
SAME	(not used)	
bytes	byte or text string(s)	byte string with concatenated bytes
BYTES	(not used)	
float	byte string (usually hex-encoded as text string)	floating point value
FLOAT	(not used)	

Table 2: App-prefix Values Defined in this Document

2.1. same: multiple literals to be checked against each other

The "same" application extension receives a sequence of one or more data items and throws an error if these data items are not the same.

Example:

```
=============== NOTE: '\' line wrapping per RFC 8792 ================

$ edn-abnf -afloat,same -e "same<< float'47110815', 37128.08203125, \
                                                    0x1.22102ap+15 >>"
37128.08203125
$ edn-abnf -afloat,same -e "same<< float'47110815', 37128.08203126 >\
                                                                   >"
** cbor-diagnostic: same<<>>: 37128.08203125 not same as 37128.\
                                                  08203126: Argument Error
$ edn-abnf -asame -e "same<< h'a10101', <<{/alg/ 1: 1 /AES-GCM 128 /\
                                                                 }>> >>"
h'A10101'
$ edn-abnf -asame -e "same<<1>>" # trivially true
1
```

2.2. bytes: Reinterpret text string as byte string

The "bytes" application extension receives a sequence of zero or more strings (throwing an error if any of these data items are not text or byte strings), concatenates their byte content and yields the concatenation as a byte string.

Examples:

```
$ edn-abnf -abytes -e 'bytes<<>>'
h''
$ edn-abnf -abytes -e 'bytes‘text1‘'
h'7465787431'
$ edn-abnf -abytes -e 'bytes<<"1", "2">>'
h'3132'
$ edn-abnf -abytes -e 'bytes<<"辰", h'"'2f'"'>>'
h'C3A42F'
$ edn-abnf -abytes -e 'bytes<<"辰", h'"'2f'"'>>' | diag2diag.rb -tu
'辰/'
```

2.3. float: IEEE754-oriented literals for more floating point values

The "float" application extension enables the notation of 2-byte, 4-byte, and 8-byte byte strings to express floating point values (mt=7, ai=25/26/27 respectively) by giving their IEEE 754 representation. A text string used as an argument is interpreted exactly as a hex literal (like the h application prefix); the result is used as the byte string.

The application-oriented literal is interpreted as an encoded data item would be that prefixes the byte string by a single byte 0xF9 (2 bytes, i.e., binary16), 0xFA (4 bytes, i.e., binary32), and 0xFB (8 bytes, i.e., binary64), respectively. Byte strings of a different length than 2, 4, or 8 raise an error.

Example:

```
=============== NOTE: '\' line wrapping per RFC 8792 ================

$ edn-abnf -afloat -e "[float'fe00', float'47110815']" -tpretty
82             # array(2)
   F9 FE00     # primitive(65024)
   FA 47110815 # primitive(1192298517)
$ edn-abnf -afloat,same -e "same<< float'47110815', 0x1.22102ap+15 >\
                                                                   >"
37128.08203125
```

The purpose of this application extension is to close a gap in EDN's [IEEE754] binary64 support: Without this (or a similar) extension there is no way to represent NaN values different from the one called out at the end of Section 4.1 of RFC 8949 [STD94]: "(for many applications, the single NaN encoding 0xf97e00 will suffice)". For finite floating point numbers, the decimal or hex floating point representations are preferred.

2.4. tbd b32/h32/c32?: Create byte string from base32 representation

// TO BE DONE: define, possibly beyond the two variants defined in [RFC4648]; watch [I-D.josefsson-rfc4648bis].

2.5. tbd b45: Create byte string from base45 representation

// TO BE DONE: define based on [RFC9285]

3. Implementation Status

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

This section records the status of known implementations of the protocol defined by this specification at the time of posting of this Internet-Draft, and is based on a proposal described in [RFC7942]. The description of implementations in this section is intended to assist the IETF in its decision processes in progressing drafts to RFCs. Please note that the listing of any individual implementation here does not imply endorsement by the IETF. Furthermore, no effort

has been spent to verify the information presented here that was supplied by IETF contributors. This is not intended as, and must not be construed to be, a catalog of available implementations or their features. Readers are advised to note that other implementations may exist.

According to [RFC7942], "this will allow reviewers and working groups to assign due consideration to documents that have the benefit of running code, which may serve as evidence of valuable experimentation and feedback that have made the implemented protocols more mature. It is up to the individual working groups to use this information as they see fit".

3.1. Implementation 1

The "float" application extension is implemented in https://cbor.me and can be enabled (afloat) in the cbor diagnostic tools (cbor-diag gem) and the edn-abnf gem.
At the time of writing, the "same" and "bytes" application extensions (asame, abytes) are only available in the gems.

3.2. Implementation 2

The float application extension is implemented in the JavaScript tools that come with the CBOR test vectors project https://github.com/cbor-wg/cbor-test-vectors/blob/main/check/files.test.js.

4. Security considerations

The security considerations of [I-D.ietf-cbor-edn-literals] apply.

// TO BE DONE: List any specific security considerations that apply to specific application extensions.

5. IANA considerations

// TO BE DONE:

6. References

6.1. Normative References

[BCP14] Best Current Practice 14, <https://www.rfc-editor.org/info/bcp14>. At the time of writing, this BCP comprises the following:

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>.

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>.

[I-D.ietf-cbor-edn-literals]
Bormann, C., "CBOR Extended Diagnostic Notation (EDN)", Work in Progress, Internet-Draft, draft-ietf-cbor-edn-literals-22, 6 April 2026, <https://datatracker.ietf.org/doc/html/draft-ietf-cbor-edn-literals-22>.

[I-D.josefsson-rfc4648bis]
Josefsson, S., "The Base16, Base32, and Base64 Data Encodings", Work in Progress, Internet-Draft, draft-josefsson-rfc4648bis-01, 11 October 2025, <https://datatracker.ietf.org/doc/html/draft-josefsson-rfc4648bis-01>.

[IANA.cbor-diagnostic-notation]
"*** BROKEN REFERENCE ***".

[IEEE754] IEEE, "IEEE Standard for Floating-Point Arithmetic", IEEE Std 754-2019, DOI 10.1109/IEEESTD.2019.8766229, <https://ieeexplore.ieee.org/document/8766229>.

[RFC4648] Josefsson, S., "The Base16, Base32, and Base64 Data Encodings", RFC 4648, DOI 10.17487/RFC4648, October 2006, <https://www.rfc-editor.org/rfc/rfc4648>.

[RFC7405] Kyzivat, P., "Case-Sensitive String Support in ABNF", RFC 7405, DOI 10.17487/RFC7405, December 2014, <https://www.rfc-editor.org/rfc/rfc7405>.

[RFC9285] Fltstrm, P., Ljunggren, F., and D.W. van Gulik, "The Base45 Data Encoding", RFC 9285, DOI 10.17487/RFC9285, August 2022, <https://www.rfc-editor.org/rfc/rfc9285>.

[RFC9741] Bormann, C., "Concise Data Definition Language (CDDL): Additional Control Operators for the Conversion and Processing of Text", RFC 9741, DOI 10.17487/RFC9741, March 2025, <https://www.rfc-editor.org/rfc/rfc9741>.

[STD68] Internet Standard 68,
<https://www.rfc-editor.org/info/std68>.
At the time of writing, this STD comprises the following:

Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, DOI 10.17487/RFC5234, January 2008, <https://www.rfc-editor.org/info/rfc5234>.

[STD94] Internet Standard 94,
<https://www.rfc-editor.org/info/std94>.
At the time of writing, this STD comprises the following:

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/info/rfc8949>.

6.2. Informative References

[RFC7942] Sheffer, Y. and A. Farrel, "Improving Awareness of Running Code: The Implementation Status Section", BCP 205, RFC 7942, DOI 10.17487/RFC7942, July 2016, <https://www.rfc-editor.org/rfc/rfc7942>.

List of Tables

Table 1: Additional EDN application extensions defined in this document
Table 2: App-prefix Values Defined in this Document

Acknowledgements

Author’s Address

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